



Universidade Federal do Rio Grande do Norte
Centro de Ciências Exatas e da Terra
Departamento de Informática e Matemática Aplicada
Laboratório de Lógica e Inteligência Computacional
Ciências da Computação

Modelagem Conceitual como Ferramenta para o Desenvolvimento de Sistemas Computacionais

Lyrene Fernandes da Silva

lyrene@dimap.ufrn.br

Orientadora: Prof^a. Dr^a. Márcia de P. B. Gottgroy

Natal – RN
Agosto - 1999

Lyrene Fernandes da Silva

lyrene@dimap.ufrn.br

Modelagem Conceitual como Ferramenta para o Desenvolvimento de
Sistemas Computacionais

Monografia apresentada ao
Departamento de Informática e
Matemática Aplicada como parte dos
requisitos para a obtenção do título de
Bacharel em Ciências da Computação.

*Para meus pais, Edilson e Graça
Que empenharam suas vidas na
realização de meus ideais.*

Agradecimentos

A Deus, que permitiu que eu conhecesse e convivesse com pessoas que contribuem diariamente para minha felicidade e sucesso.

Meus pais e irmãos pelos 23 anos de apoio, incentivo e amor.

A professora Márcia Gottgroy, que me orientou, incentivou e confiou em meu trabalho.

Aos meus amigos: Fernanda, Andréa, Marília, João, Vécio, Ítalo, Guedes, César, Giordano, José Maria e toda a turma de 94.1 que sempre me fizeram morrer de rir mesmo nas horas de angústia no decorrer deste curso.

A Leandro, meu namorado, pelo carinho e paciência durante a edição deste relatório.

Resumo

Diante da atual situação do desenvolvimento tecnológico, o mundo comercial das organizações e o mundo científico dedicam-se ao desenvolvimento de novas metodologias e técnicas que dêem suporte ao desenvolvimento de sistemas computacionais mais flexíveis, portáteis e confiáveis. Dentro deste contexto, este relatório aborda duas perspectivas para análise e design de sistemas baseadas no paradigma conceitual, a modelagem estática-dinâmica e a UML (Unified Modeling Language), identificando suas características, vantagens, desvantagens e aplicabilidade. O estudo teórico destes métodos foi acompanhado de um estudo de caso para experimentação do conhecimento adquirido, constituindo na modelagem do SAGRI – Sistema de Apoio a Produção Agrícola em ambos os modelos.

Abstract

Before the current situation of the technological development, the commercial world of the organizations and the scientific world they are devoted to the development of new methodologies and techniques that give support to the development of systems more flexible computacionais, you carried and you trusted. Inside of this context, this report approaches 2 perspectives for analysis and design of systems based on the conceptual paradigm, the modelling static-dynamics and UML (Unified Modeling Language), identifying your characteristics, advantages, disadvantages and aplicabilidade. The theoretical study of these methods was accompanied of a case study for experimentation of the acquired knowledge, constituting in the modelling of SAGRI–System of Support the Agricultural Production in both models.

Sumário

1 - INTRODUÇÃO	8
2 – MODELAGEM CONCEITUAL	11
2.1 – EVOLUÇÃO DOS PARADIGMAS	11
2.2 – ÁREAS CORRELATAS	14
2.2.1 – Modelos Mentais	14
2.2.2 – Teoria de David Ausubel	16
2.3 – CONCEITOS E CARACTERÍSTICAS DO MODELO CONCEITUAL	18
2.4 – CONSTRUINDO UM MODELO CONCEITUAL	19
2.5 – GRANDES GRUPOS DE CONHECIMENTO MODELADOS PELO MODELO CONCEITUAL	20
2.6 – NOTAÇÃO	21
2.7 – APLICABILIDADE	22
3 – MODELAGEM ORIENTADA A OBJETOS ATRAVÉS DA UML	23
3.1 – O MODELO ORIENTADO A OBJETOS	23
3.2 – UML (UNIFIED MODELING LANGUAGE)	24
3.3 – DIAGRAMAS DA UML	25
3.3.1 – Diagrama de Casos de Uso	25
3.3.2 – Diagrama de Classes	26
3.3.3 – Diagrama de Objetos	26
3.3.4 – Diagrama de Interação	27
3.3.5 – Diagrama de Estados	28
3.3.6 – Diagrama de Atividades	28
3.3.7 – Diagrama de Implementação	28
3.4 – UTILIZAÇÃO DA UML	29
3.4.1 – Fases do Desenvolvimento de um Sistema Computacional	29
4 – MODELAGEM DO SISTEMA	32
4.1 – O SISTEMA SAGRI	32
4.2 – DESCRIÇÃO DO PROBLEMA	35
4.3 – MODELAGEM CONCEITUAL	36
4.3.1 – Definição do Problema	37
4.3.2 – O Modelo Dinâmico	37
4.3.3 – O Modelo Estático	41
4.4 – MODELAGEM ORIENTADA A OBJETOS ATRAVÉS DA UML	44
5 – CONCLUSÕES OBTIDAS COM AS MODELAGENS	59
5.1 - MODELAGEM EM UML A PARTIR DO MUNDO REAL	60
5.2 - MODELAGEM EM UML A PARTIR DO MODELO CONCEITUAL	60
6 – CONCLUSÕES E PROJETOS FUTUROS	62
7 – BIBLIOGRAFIA	63
APÊNDICE A: NOTAÇÃO DA UML	66
APÊNDICE B: MODELAGEM CONCEITUAL	77
APÊNDICE C: MODELAGEM ORIENTADA A OBJETOS ATRAVÉS DA UML	93

1 - Introdução

“Uma vez que os computadores se tornam cada vez mais complexos, os seres humanos devem desistir de pensar como eles. Em vez disso, os computadores devem ser feitos de forma a pensar como seres humanos” (James Martin & James Odell).

Esta idéia tem motivado e levado o mundo comercial e o mundo científico a dedicarem-se no desenvolvimento de sistemas computacionais, que sejam cada vez mais naturais ao ser humano.

Temos assistido à transformação não só das linguagens, como por exemplo da linguagem Assembler para linguagens de alto nível como C ou Pascal ou ainda para linguagens visuais orientadas a objetos como, também, à transformação de toda uma metodologia e filosofia que há por trás destas linguagens pois, o que era baseado em processos hoje se baseia em conceitos.

Tal mudança de pensamento surgiu para suprir as necessidades de um mercado que precisa cada vez mais da automação de suas atividades, exigida de maneira ordenada e atendendo a requisitos básicos de qualidade, como por exemplo flexibilidade, portabilidade, correteude, confiabilidade etc.

Para atingir aos requisitos de qualidade exigidos pelo mercado são necessários modelos com maior abrangência no que diz respeito ao conhecimento envolvido no sistema. Torna-se cada vez mais evidente a existência de um salto não muito pequeno entre o mundo real e os modelos computacionais, o que acarreta uma perda considerável de conhecimento entre os dois modelos e, conseqüentemente, uma distorção do mundo real em termos computacionais.

Este cenário tem motivado a busca por modelos e metodologias que garantam uma transformação do conhecimento de forma mais gradativa, ou seja, sem grandes discrepâncias, que proporcionem uma retratação o mais fiel possível do mundo real em um mundo artificial (em um modelo computacional). Estes modelos podem ser considerados como um passo a mais na evolução dos paradigmas de representação do conhecimento.

1.1 - Justificativas e Objetivos

O Sagri – Sistema Inteligente de Apoio a Produção Agrícola, é um sistema que se propõe a auxiliar o agricultor na obtenção de maior produtividade, menos perdas e custos e um melhor aproveitamento dos recursos naturais [1][2][3][4][5]. Baseia-se em conhecimento e comporta várias tecnologias em seu contexto funcional como, por exemplo, bases de conhecimento, banco de dados, inferência fuzzy, redes neurais etc.

A complexidade do sistema Sagri estende-se às tecnologias envolvidas no mesmo e ao próprio domínio do conhecimento. Sendo assim, se faz necessário um maior cuidado durante

o seu desenvolvimento, ou seja, um bom gerenciamento deste processo para se obter como resultado um sistema com um bom padrão de qualidade.

Os cuidados a serem tomados vão desde a aquisição do conhecimento até a implementação do sistema computacional. No entanto a aquisição é uma fase de maiores riscos pois, todo o restante do projeto se baseará nela. É preciso, nesta fase, construir modelos mais expressivos do conhecimento, seja a partir de livros, de documentos ou adquirido dos especialistas, com o mínimo de perda possível, para só então especificá-lo em um formalismo de representação como, por exemplo, a orientação a objetos, grafos, especificação formal etc. Desta forma, a utilização destes modelos permitirá o emprego de tecnologias que se adequem ao sistema ou a parte dele e não vice-versa, minimizando enormemente as perdas do conhecimento a ser implementado.

Vale esclarecer que esta não é uma necessidade exclusiva do Sagri mas, de todo e qualquer sistema que tenha um mínimo de complexidade, característica das tarefas concretas do dia a dia das organizações ou centros de pesquisa, que seja desenvolvido por uma equipe e que se proponha a ter uma vida útil relativamente longa o que, no momento, é absolutamente desejável.

Atualmente, a tecnologia que mais está apta a subsidiar o desenvolvimento de sistemas computacionais é a Orientação a Objetos (O. O.) pois, fornece recursos que abrangem todas as fases de desenvolvimento, desde a análise até a implementação, além de ser a técnica compreensível pela máquina, menos artificial ao homem. Por estes motivos, a Orientação a Objetos e mais especificamente a UML (Unified Modeling Language), também serão abordadas neste trabalho.

Diante da crescente complexidade dos sistemas atuais, um trabalho minucioso e efetivo de modelagem deve ser realizado. Há a necessidade de percorrer cada etapa do processo de transformação de conhecimento, do modelo concreto ao modelo computacional, de forma a garantir o mínimo de perdas, objetivando construir sistemas mais reais e de melhor desempenho [6].

O objetivo deste relatório é descrever os métodos conceituais, o estático-dinâmico¹ e o orientado a objetos proposto pela UML, de forma que seja viável, analisá-los enquanto ferramenta para o desenvolvimento dos sistemas computacionais atuais, inteligentes.

Esta análise é realizada através da avaliação das características de cada método, descritas pela teoria que os embasa e de um estudo de caso do sistema SAGRI. O estudo de caso utiliza a modelagem conceitual do sistema Sagri como base para o modelo orientado a objetos de forma que se possa verificar o quanto é necessário um modelo intermediário entre o mundo real e o modelo técnico orientado a objetos.

1.2. Organização do Texto

O presente texto está dividido em 5 capítulos.

Neste primeiro capítulo apresentou-se o contexto em que está inserido o trabalho, descrevendo o que motivou a trabalhar sobre o tema e precisamente qual seu objetivo.

¹ O modelo conceitual estático-dinâmico é chamado normalmente, neste trabalho, de Modelo Conceitual pois denota o modelo puramente conceitual, conceitual em sua essência.

O Capítulo 2 apresenta o resultado do estudo teórico sobre Modelos Conceituais como ferramenta para o desenvolvimento de sistemas computacionais.

O Capítulo 3 apresenta o resultado do estudo da modelagem orientada a objetos através da UML. Nos capítulos 2 e 3 são identificadas as características, vantagens, desvantagens e aplicabilidade dos dois modelos.

O Capítulo 4 descreve o estudo de caso realizado, a modelagem concebida e as conclusões obtidas com a experimentação.

O Capítulo 5 apresenta as conclusões obtidas sobre o estudo realizado, e aponta outros trabalhos que podem ser desenvolvidos sobre o tema.

2 – Modelagem Conceitual

2.1 – Evolução dos Paradigmas

É intrínseco ao ser humano o desejo, e mesmo a necessidade de resolver os problemas que os rodeia. Buscando sempre uma solução mais eficiente, o ser humano chegou ao nível de evolução que se encontra hoje, e encontrou na computação a melhor forma para solucionar muitos dos seus problemas cotidianos, sejam eles simples ou complexos.

Este avanço tecnológico permite que hoje se busquem soluções computacionais para problemas de alta complexidade até então não confiáveis às máquinas ou nem mesmo possíveis de serem implementados.

Além de ocorrer devido a sua eficácia como ferramenta de apoio e conseqüente credibilidade das pessoas, a evolução computacional ocorre também, e principalmente, devido à evolução humana, caracterizada principalmente pela crescente elaboração de seus modelos mentais. Evolução esta que impulsiona a evolução computacional como forma de satisfazer as necessidades surgidas ou amplificadas com a sua própria evolução.

Tais necessidades vão desde uma simples edição de texto, planilhas, panfletos, etc, bem formatados, até o comércio eletrônico ou ensino à distância, ou ainda diagnósticos médicos ou realização de tarefas de grande risco, enfim há inúmeras aplicações de grande complexidade, hoje, efetivadas por computadores de maneira satisfatória.

Vários conceitos caracterizadores dos sistemas computacionais ou não-computacionais também sofreram inúmeras transformações como, por exemplo, o conceito de “qualidade”. Nos sistemas computacionais deixou-se de se preocupar apenas com a realização de tarefas e passou-se a tentar efetiva-las da melhor forma possível. Esta “melhor forma” de efetuar tarefas proporcionou uma reviravolta na concepção de sistemas, deixou-se de ter uma visão centralizadora do mundo e dos processos. Passou-se a valorizar mais o usuário e como ele enxerga o mundo, o que deseja e necessita, para que se possa construir sistemas mais “amigáveis” e realmente úteis.

Assim qualidade de sistemas computacionais requer hoje [7]: Flexibilidade, Portabilidade, Reusabilidade, Interoperabilidade, Corretude, Confiabilidade, Eficiência, Integridade e Usabilidade. Tais características estão em contínuo aperfeiçoamento e necessitam que o processo de engenharia dos sistemas evolua para que possam ser plenamente satisfeitos.

A engenharia de software muito tem contribuído para construção de softwares de qualidade, propondo ciclos de vida, exigindo uma boa documentação que auxilia na etapa de manutenção, criando procedimentos como a reengenharia, e propondo um gerenciamento do projeto. No entanto, a grande maioria desses processos, embora cite, não dão ênfase inclusive a disponibilização de ferramentas que suportem à análise do “sistema real” sobre o qual o

sistema computacional vai trabalhar ou simular, focando já mais a frente à análise do sistema computacional em si. Ou seja, parte do pré suposto que a análise do sistema real já tenha sido realizada, e que as melhores soluções para os problemas já tenham sido encontradas.

Um sistema computacional pode ser considerado como sendo um modelo implementado em computador para resolução de algum problema do mundo real. Este modelo computacional consiste de uma representação do mundo e de um mecanismo de solução, ambos computacionalmente representados. Atualmente esta representação retrata o modelo mental que o projetista possui do problema a ser solucionado, influenciado, assim, por suas crenças, intenções e preferências [8][9], podendo desta forma ser um modelo completo ou incompleto, certo ou errado. E sem que se tenha feito um estudo mais elaborado do sistema real e uma análise mais rigorosa das características do sistema, é feita a representação computacional. Por ser um retrato do modelo mental, os modelos computacionais possuem as mesmas características daqueles modelos, sejam elas boas ou ruins.

Percebe-se que existe uma transformação do conhecimento sobre o mundo no decorrer do processo de construção dos sistemas computacionais. Tal transformação, infelizmente não se dá apenas no mapeamento da “linguagem” de um modelo na “linguagem” de outro modelo. Ela se dá, também, com relação as características de cada modelo. Ocorre uma grande perda de informações no decorrer deste processo a começar pela linguagem humana que, por ser ambígua e dependente de contexto, não permite descrever o mundo tal como ele é, passando pelo modelo mental que também é apenas um dos inúmeros modelos possíveis de serem abstraídos do mesmo contexto, e terminando pelos modelos e linguagens computacionais que são enormemente limitados no que diz respeito à sua expressividade. Estas perdas são responsáveis por sistemas inadequados ou ineficientes que, muito possivelmente, não atenderão aos usuários satisfatoriamente e os resultados colhidos do mercado mostram que geralmente acabam em estado de desuso.

Há a necessidade de se ter modelos intermediários entre o mental e o computacional. Modelos que abram espaço para análise e questionamento do problema em pauta e do próprio modelo mental do projetista, onde se permita fazer uma retratação do sistema real. Desta forma se obterá sistemas computacionais mais fiéis ao mundo real, ou seja, mais aptos de gerar soluções efetivas e úteis. Seguindo esta linha de raciocínio, a Figura 1 [10] mostra a trajetória que o conhecimento deve seguir até ser implementado, este ciclo de vida é utilizado como base para a Metodologia KADS² [10].

² A metodologia KADS foi desenvolvida na Universidade de Amsterdã em 1983 como resultado do projeto ESPRIT – European Community Initiative for Research into Technology para dar suporte ao desenvolvimento de sistemas baseados em conhecimento.

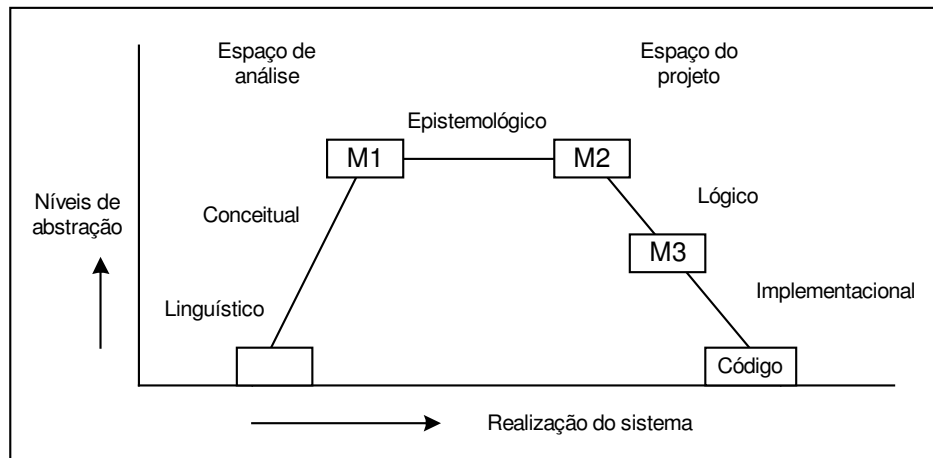


Figura 1: Etapas do Desenvolvimento de Sistemas Computacionais

O desenvolvimento de sistemas, como foi exposto, baseia-se num processo de transformação, onde, segundo os princípios da metodologia KADS, podem ser identificados basicamente 5 níveis: Linguístico, Conceitual, Epistemológico, Lógico e Implementacional. Estes níveis compreendem 2 fases, a do Espaço de Análise e a do Espaço de Projeto. A primeira referente à aquisição de conhecimento e formulação de um processo de solução para resolução do problema proposto e a segunda, ligada à concretização do modelo de implementação, no caso em estudo, do modelo computacional do sistema. Como pode ser observado, os níveis Linguístico e Conceitual, compreendidos na primeira fase, são a base para se alcançar os demais níveis.

O nível Linguístico retrata o mundo real; através dele se adquire todo o conhecimento referente ao problema que se quer resolver, buscando conhecer todo o ambiente relacionado ao domínio de interesse. O modelo linguístico é a forma de expressão do modelo mental, ou seja, a visão, o entendimento do problema concreto, entendimento de alguém: usuário, gerente, técnico ou especialista. Esse modelo é a fonte de material que o Analista de Sistemas ou Engenheiro de Conhecimento possuem para compreender e mapear o problema em questão. Para trabalhar este modelo são aplicadas técnicas de aquisição de conhecimento como por exemplo, entrevistas, discussão dirigida, elicitação construtiva, etc [5] [11].

No nível Conceitual, tenta-se mapear todo o conhecimento adquirido, em vários níveis de abstração, partindo de um menos detalhado, no entanto mais genérico, meta, para o mais detalhado possível, mais específico de dados e de situações. Busca-se ter “uma vista aérea” do problema para que se possa visualizar o Modelo de Solução mais adequado e a partir daí, a forma “mais adequada” de implementação do sistema computacional.

No nível Epistemológico, ocorrerá a transição da representação do mundo real para a representação do mundo computacional. Espera-se, neste estágio, que o Modelo do Problema construído, a partir dos modelos conceituais, forneça ao engenheiro todos os subsídios de que ele necessita para apontar o Modelo de Solução que melhor se adequa à implementação do problema, conduzindo dessa forma a transformação, gradativa, dos modelos conceituais nos modelos lógicos, propiciando o mínimo de perdas do conhecimento adquirido. Este é o nível de mais alta abstração, a partir dele se inicia a segunda fase do desenvolvimento dos sistemas, o Espaço do Projeto.

No nível Lógico, é realizada a representação dos conceitos modelados, aplicando-se os formalismos de representação do conhecimento, como por exemplo, Orientação a Objetos, Especificação Formal, Grafos, Redes Semânticas, Regras de Produção, dentre outros, o que mais se adequar à resolução do problema, de acordo com o Modelo de Solução adotado.

No nível Implementacional é onde ocorre, finalmente, a codificação do modelo de solução na linguagem de máquina, ou seja, de forma que o conhecimento representado ao longo do desenvolvimento do modelo possa ser manipulado pela máquina [8], caracterizando, assim, o modelo computacional do problema proposto. Este nível finaliza o processo de desenvolvimento de um sistema computacional.

Ao término de cada uma destas fases, tem-se um modelo que representa o mundo real, porém o modelo conceitual é o modelo mais natural ao ser humano, descritivo do modelo mental e estrutura cognitiva do mesmo, retrata, assim, o mundo da maneira mais real possível, sem se prender a limitações impostas pelos formalismos das tecnologias existentes.

Até há bem pouco tempo, a maioria dos projetistas partiam do modelo linguístico, faziam um breve modelo conceitual e iam direto para a fase implementacional. O avanço tecnológico tem sido impulsionado e concretamente tem gerado “ferramentas”, cada vez melhores por esse caminho de transformação (apresentado na Figura 1), gerando a aproximação do modelo implementacional do concreto. Atualmente com o advento da orientação a objetos, das linguagens lógicas, principalmente as não clássicas e linguagens funcionais, permitiu-se a construção mais efetiva dos modelos lógicos, já que estas visões permitem uma manipulação de conceitos mais semelhantes com os que se está habituado a tratar no mundo real, além de oferecer uma maior flexibilidade na implementação, se comparado com o que se tinha há algum tempo (apenas as linguagens) .

No entanto, embora os modelos lógicos tratem da representação computacional do mundo e da solução de um certo problema, ainda não é o suficiente pois, para chegar a este estágio é necessário uma análise detalhada do mundo e a construção de soluções efetivas e eficientes antes de se pensar numa representação computacional, ou seja, cada vez mais se faz necessário caminhar pelo nível conceitual e pelo nível epistemológico de maneira efetiva, para que se obtenha modelos computacionais realmente eficientes.

2.2 – Áreas Correlatas

O modelo conceitual é um modelo artificial dos processos de raciocínio do ser humano e tem suas origens na Filosofia, Psicologia e áreas correlatas com a cognição. Estas áreas influenciam a Computação com seus estudos sobre inteligência, processo de aprendizagem e outras já que a informática vive em função de satisfazer os usuários da maneira mais ampla possível [12][13].

2.2.1 – Modelos Mentais

“Modelo mental é o modelo que as pessoas têm de si mesmas, dos outros, do ambiente e das coisas com as quais interagem. As pessoas formam modelos mentais através de experiências, treinamento, e instrução”

- Donald Norman [12]–

“Modelo mental é uma representação interna de informações que correspondem analogamente ao estado de coisas que estiver sendo representado, seja qual for ele. Modelos mentais são análogos estruturais do mundo”

- Marco Antônio Moreira [12]–

“Modelos mentais são os mecanismos através dos quais os humanos são capazes de gerar descrições do propósito e forma de um sistema, explicar o funcionamento de um sistema e os seus estados observados e prever os estados futuros”

- Rouse e Morris [13] -

A partir das definições acima, percebe-se que um modelo mental é um modelo existente na mente de cada indivíduo. Ele reflete o que se pensa sobre alguma coisa em uma certa circunstância. Desta forma, um mesmo indivíduo pode construir modelos distintos para o mesmo objeto (coisa, pessoa ou ambiente) pois, estes serão coerentes com o uso a que se deseja empregar o objeto[14].

O modelo mental é completamente pessoal, baseado no conhecimento, nas crenças e nos desejos de cada ser. Eles servem para explicar o comportamento de sistemas, fazer previsões, localizar falhas e atribuir causalidade aos eventos e fenômenos observados. São adquiridos através de transmissão cultural ou ensino, interação cotidiana com outras pessoas e com o mundo [13]. São limitados pelo conhecimento, experiência e pela própria estrutura do sistema de processamento de informação humano. Podem conter elementos desnecessários, errôneos ou contraditórios. É robusto o suficiente para permitir imprecisões, ou falta de informações.

Este tema tem sido de grande interesse para os pesquisadores de informática. Os modelos mentais estão sendo largamente estudados e aplicados, principalmente em projetos de interface pois, neles é necessário conhecer o que os usuários sabem, e de que maneira pensam a respeito de como os sistemas funcionam, para poder se construir sistemas mais amigáveis de fácil aprendizado e de maior facilidade para desempenhar tarefas. Um estudo realizado sobre os Helps dos programas [15] mostrou que eles são muitas vezes ineficientes principalmente por haver um distanciamento bastante considerável entre os modelos mentais do usuário e os modelos prescritos pelos projetistas para os usuários e mais ainda, entre os modelos mentais dos usuários e os dos projetistas.

De uma maneira mais abrangente, os modelos mentais têm sido concretamente enfocados na linha de pesquisa e desenvolvimento de Inteligência Artificial Distribuída, dando base ao surgimento do novo paradigma de desenvolvimento de sistemas computacionais, o paradigma baseado em Agentes.

Assim, o ser humano continua evoluindo e não almeja fazer com que toda a humanidade aprenda a pensar (construir modelos mentais) tecnicamente, como os projetistas pensam, mas sim o contrário (embora isto ocorra em ambos os sentidos). Deseja-se desenvolver a tecnologia de tal maneira que os projetistas do futuro não precisem possuir “modelos mentais técnicos”.

2.2.2 – Teoria de David Ausubel

David Ausubel propõe uma explicação teórica do processo de aprendizagem, segundo um ponto de vista cognitivista [16]. Como conseqüência desenvolveu uma ferramenta para facilitar e tornar o mecanismo da aprendizagem efetivo para os aprendizes³.

Segundo Ausubel, a aprendizagem é um processo de armazenamento de informações. Incorporação à uma estrutura no cérebro do indivíduo de modo que esta possa ser manipulada e utilizada no futuro. Neste contexto, a habilidade de organização das informações deve ser desenvolvida.

Para Ausubel, aprendizagem significa organização e integração do material na estrutura cognitiva. Ele considera que o fator mais importante a influenciar à aprendizagem é aquilo que o aprendiz já sabe.

A estrutura cognitiva é formada por conceitos, idéias e proposições, também chamadas por Ausubel de subsunçores, que servem de ancoradouro à novas informações, permitindo que o aprendiz atribua-lhes significado.

A medida que a aprendizagem começa a se tornar *significativa*, esses subsunçores vão se tornando cada vez mais elaborados e mais capazes de servir de ancoradouro a novas informações.

O processo de subsunção também chamado por Ausubel de princípio da assimilação é descrito da seguinte maneira (Figura 2 [16]):

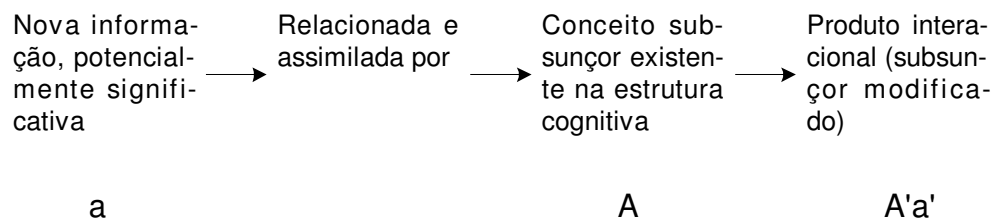


Figura 2: Processo de subsunção

A nova informação a e o conceito subsunçor A são modificados pela interação entre eles, resultando em A'a'. Mais tarde este novo conceito se tornará não dissociável, constituindo um subsunçor mais bem elaborado.

Há, assim, uma expansão e redução da estrutura cognitiva no decorrer da aprendizagem cujo resultado é a retenção de idéias, conceitos e proposições mais gerais e estáveis (meta conhecimento), de onde pode-se inferir conceitos específicos.

Como ferramenta facilitadora da aprendizagem significativa em sala de aula, Ausubel propõe o que se chama de Mapas Conceituais, que são representações gráficas semelhantes a diagramas que descrevem a estrutura de uma certa disciplina de forma ordenada e seqüencial

³ O termo aprendiz é usado aqui no sentido geral de o “ser que aprende” e não no sentido específico de “aquele que aprende arte ou ofício”.

[16]. Desta maneira, o professor auxilia o aluno a assimilar a estrutura das disciplinas e a reorganizar sua própria estrutura cognitiva.

“Um mapa conceitual é um dispositivo que usa a teoria de Ausubel para permitir a construção de material para aprendizes que possuem diferentes conhecimentos anteriores”

- Apuena Gomes [10] -

Estes diagramas são construídos seguindo a hierarquia dos conceitos. Geralmente os conceitos mais gerais aparecem no topo do mapa, e seguem uma ordem descendente até os conceitos mais específicos na base do mapa.

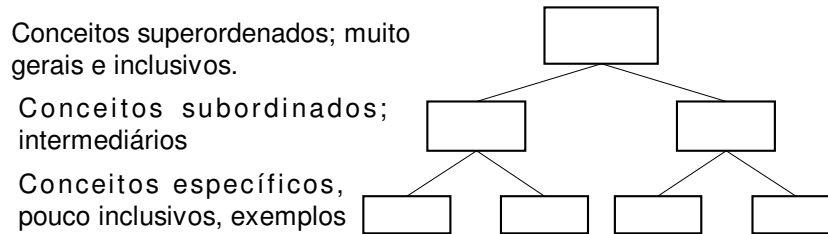


Figura 3: Modelo simplificado de um mapa conceitual

Um modelo simplificado de um mapa conceitual é apresentado na Figura 3 [16]. Há várias maneiras de se conduzir a explicação de disciplinas utilizando mapas conceituais, duas delas são a visão de Ausubel e a visão de Novak:

Segundo a visão Ausubeliana [16] o desenvolvimento de conceitos procede-se melhor quando os elementos mais gerais são introduzidos primeiro e é progressivamente diferenciado, em termos de detalhes e especificidade.

A visão de Novak [16] defende que é necessário “subir” e “descer” na estrutura conceitual a medida que a nova informação é apresentada.

Como foi visto, um mapa conceitual pode ser construído de diversas maneiras, ou seja, não há um único mapa para determinada disciplina, e sim quantos forem desejáveis. E, além disso, um mesmo mapa conceitual pode ser manipulado de maneiras diferentes, pela visão de Ausubel ou pela visão de Novak.

Dado o que foi exposto, pode-se considerar que um mapa conceitual é um modelo conceitual, que embora construído sob a influência dos modelos mentais do construtor, permite que se faça uma análise da disciplina e de seus próprios modelos mentais, possibilitando a eliminação de inconsistências, erros e dúvidas e uma aprendizagem e posterior aplicação do modelo (tanto o conceitual quanto o mental).

A teoria de Ausubel com seus mapas conceituais, despertaram o interesse dos profissionais de informática. Esta abordagem é capaz de oferecer aos analistas de sistemas, recursos que auxiliam desde a aquisição até a representação de conhecimento, justamente auxiliando a galgar a “subida conceitual”, permitindo analisar melhor o contexto focado pelo sistema a ser implementado, sendo ferramenta efetiva no desenvolvimento de sistemas com capacidade de aprendizado.

2.3 – Conceitos e Características do Modelo Conceitual

“Modelo conceitual é um modelo de uma área de uma organização, o qual não envolve detalhes de implementação e descreve tanto propriedades estáticas quanto propriedades dinâmicas do sistema modelado.”

- Douglas José P. Azevedo [17] -

“Modelos conceituais são projetados como instrumentos para compreensão ou para o ensino de sistemas físicos”

- Donald Norman [12] -

O modelo conceitual se propõe a representar o mundo real de forma sistematizada. Este tipo de representação não comporta nenhuma tecnologia, ele é completamente informal e intuitivo para o ser humano, tendo o objetivo de modelar o conhecimento tal como ele é. A importância de se construir um modelo como este está, entre outros aspectos, na necessidade de se organizar as idéias antes de implementar qualquer sistema. É necessário analisar, inicialmente, todo o domínio de interesse, as variáveis em questão, as possíveis soluções para o problema e, é claro, documentar todas estas etapas, para que não se perca de vista, em momento algum, o objetivo principal do sistema.

Com o modelo conceitual tenta-se diminuir as diferenças existentes entre o modelo mental e os modelos de representação computacional do conhecimento, permitindo que este navegue por todas as fases do ciclo de vida (figura 1), transformando-se de forma mais gradativa, com pequena ou nenhuma perda de uma fase para outra.

Os modelos conceituais, com a proposta de retratar o mundo real com fidelidade, capacita os desenvolvedores de sistemas a construir sistemas mais eficientes, dinâmicos, flexíveis e confiáveis, além de poder ajuda-los no processo de aquisição do conhecimento. Nesta fase o Modelo Conceitual deve ser construído pelos projetistas sob a orientação dos usuários detentores do conhecimento, assim, estes poderão validar as informações modeladas.

Por forçar uma organização, avaliação e validação das idéias, o Modelo Conceitual auxilia na elaboração de raciocínios, proporcionando aprendizagem sobre o sistema. Quanto maior a aprendizagem melhor será a manipulação do conhecimento adquirido. Isto possibilitará a eliminação de inconsistências, erros e falhas. Possibilitará, também, fazer previsões, servindo como ferramenta para estudos antecipatórios de comportamentos e situações, podendo ser detectados problemas antes mesmo de sua implementação. Isto fará com que se busque modelos de representação mais adequados a tratarem certo tipo de problema.

A aquisição de conhecimento tem no Modelo Conceitual um forte aliado pois, este além de capacitar o analista a descrever o conhecimento com o mínimo de perdas. Funciona, também, como meio de comunicação, tanto entre projetistas e analistas, quanto entre analistas e usuários, já que sua linguagem deve ser simples. Assim se atende tanto as necessidades dos desenvolvedores, quanto se oferece uma maior confiabilidade aos usuários pois, estes perceberão que o conhecimento transferido está sendo entendido de forma adequada, e que o sistema deverá refletir suas expectativas.

As principais características do Modelo Conceitual são:

- ✓ Simplicidade de compreensão pelo usuário do sistema e pelos projetistas;
- ✓ Indicação clara dos objetivos ou propósitos;
- ✓ Facilidade de controle, manipulação e comunicação dos resultados;
- ✓ Completude, no nível de abrangência necessária;
- ✓ Adaptativo, com procedimentos fáceis de modificação e atualização;
- ✓ Evolucionário, podendo iniciar-se simples tornando-se mais complexo em função da utilização.

Assim como os modelos mentais e mapas conceituais um Modelo Conceitual também é apenas uma das possíveis representações de um sistema específico.

2.4 – Construindo um Modelo Conceitual

A construção do Modelo Conceitual é um processo de análise e aquisição de conhecimento [18]. Ela é projetada a partir do mundo real focado por um determinado problema. Porém, vale lembrar que embora o modelo conceitual tente retratar o mundo real com fidelidade, ele está subordinado ao contexto a que se está trabalhando.

O Modelo Conceitual é construído conforme os objetivos e necessidades da análise. A construção do modelo proporciona uma maneira sistemática, explícita e eficiente dos analistas orientarem seus julgamentos e decisões. Serve como um meio conveniente de comunicação e auxílio ao raciocínio.

Existe um tipo de modelo lógico mais apropriado para cada sistema. O Modelo Conceitual auxilia na escolha deste modelo já que oferece ao seu criador uma visão abrangente e profunda do conhecimento inerente ao caso em questão. Como exemplo, no caso do SAGRI percebeu-se uma grande quantidade de conceitos imprecisos relacionados à área agrícola fazendo com que fosse adotada a lógica fuzzy para o tratamento destes conceitos [3].

No entanto, não deve haver dependência alguma entre o mapeamento e o formalismo de representação adotado, podendo, nesse caso, ser uma representação baseada tanto em lógica, como em scripts, frames, redes semânticas ou regras de produção. Isto fornece uma grande portabilidade a modelagem, sendo exatamente esta uma de suas principais características pois, o que se pretende é mapear o conhecimento sem perder o senso indutivo utilizado pela mente humana.

Não há ainda uma metodologia bem definida para a construção do Modelo Conceitual, no entanto algumas etapas foram identificadas como sendo de importância primordial para o processo de modelagem, são elas:

- ✓ Definir os objetivos. É necessário que o desenvolvedor tenha em mente os objetivos do projeto. Isto, embora pareça lógico, é uma questão que muitas vezes é esquecida ou não trabalhada de maneira apropriada, fazendo com que muitas vezes, se desperdice tempo, com pormenores dispensáveis ao sistema, detalhando demais, chegando a níveis de abstração que não seriam necessários, por exemplo.
- ✓ Delimitar o escopo. Um modelo não se destina a representar o universo, mas sim a mapear uma parte dele à luz de um objetivo definido.

- ✓ Adquirir o conhecimento necessário. Os analistas aplicarão técnicas de aquisição de conhecimento e utilizarão o próprio modelo como fonte de dúvidas e de esclarecimentos.
- ✓ Identificar as possíveis soluções para o problema. Com o decorrer da construção do modelo, soluções mais elaboradas vão se tornando visíveis ao projetista.
- ✓ Abstrair e detalhar cada vez mais. A construção de um Modelo Conceitual deve iniciar-se simples, evoluindo para uma maior elaboração, com níveis de complexidade adaptados aos objetivos propostos.

2.5 – Grandes Grupos de Conhecimento Modelados pelo Modelo Conceitual

Para o desenvolvimento de sistemas computacionais, o Modelo Conceitual deve ser capaz de exprimir diversos níveis de detalhamento do conhecimento, agrupados em dois grandes grupos de tipos de conhecimento: o conhecimento estático, conceitos-objetos envolvidos e o dinâmico, ou seja, dos conceitos-processos envolvidos. Assim os Modelos Conceituais disponibilizam aos usuários e desenvolvedores, o conhecimento necessário sobre as relações entre conceitos e processos a serem executados para serem atingidos os objetivos do sistema em concepção. A Figura 4 [18] mostra a modelagem englobando estes dois modelos.

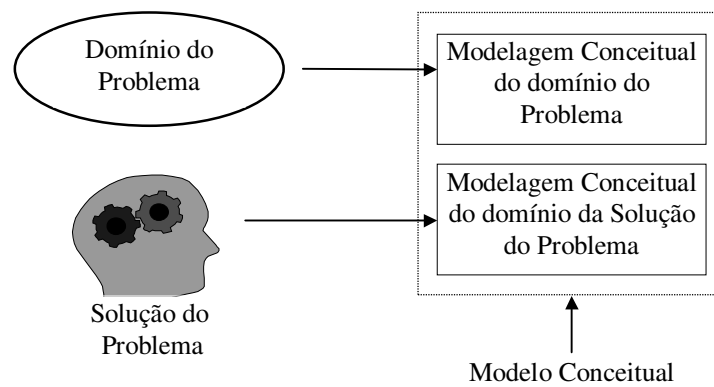


Figura 4: Modelagem conceitual: escopo e processo de solução

O modelo estático, também denominado de descritivo, descreve todo o conhecimento inerente ao sistema. Retrata “o que” o sistema deve fazer, apresentando todos os conceitos e relacionamentos entre conceitos envolvidos no escopo do problema a ser solucionado. Ele deve mostrar o ambiente envolvido tal como ele é, organizadamente, de forma a proporcionar um estudo de abrangência total sobre o mesmo.

O modelo dinâmico, também denominado de prescritivo, descreve uma solução para o problema proposto. Retrata “como” o sistema deve trabalhar para atingir seus objetivos. Permite-se analisar qual a solução mais adequada para um certo problema, utilizando o modelo estático paralelamente definido.

Ambos os modelos representam o mundo através de conceitos e relacionamentos. Os relacionamentos indicam a hierarquia existente entre os conceitos e/ou a influência que os conceitos exercem entre si.

As pessoas geralmente fazem confusão na distinção dos dois modelos, já que são dois modelos descritivos. Entretanto, deve ficar claro que um descreve o processo de solução e o outro descreve as características do sistema real. A maior fonte de confusão é que precisamos descrever conceito-processos, que são, por definição, intrinsecamente dinâmicos, distinguindo-se do processo de solução do problema.

Sendo assim, torna-se possível construir vários modelos dinâmicos utilizando o mesmo modelo estático, tanto para solucionar problemas diferentes como para solucionar o mesmo problema de maneiras diferentes.

No estudo de caso do Sagri, por exemplo, poderia se obter uma prescrição do solo a partir das limitações do solo ou a partir do valor agrônômico que ele possui. Para ambos os processos de solução, teria-se como base o mesmo modelo estático. (ver Capítulo 4)

Existem, ainda, duas maneiras de conduzir a modelagem. Uma linha acredita que se deve iniciar o processo de modelagem pelo modelo estático do problema, ainda que de maneira simplificada, já que os conceitos estão no mundo mesmo antes que o homem lhes atribua uma função.

Segundo uma outra visão deve-se partir do modelo dinâmico, já que não há interesse em se representar o mundo real sem pensar em se resolver algum problema.

Em ambos os casos, após a iniciação deve-se proceder construindo o modelo estático e o dinâmico paralelamente, de maneira que os conceitos identificados no estático sejam utilizados no dinâmico e os conceitos identificados no dinâmico sejam posteriormente, se necessário, acrescentados ao modelo estático.

Dependendo da complexidade do sistema, aconselha-se que sejam construídos modelos isolados para conceitos de grande abrangência, e para que a visibilidade do modelo não seja dificultada, uma vez que outra característica fundamental do Modelo Conceitual é sua representação gráfica para facilitar o entendimento do processo.

2.6 – Notação

Conceitos: os conceitos são representados por elipses contendo em seu interior o nome do conceito.

Relacionamentos: os relacionamentos são representados por linhas direcionadas e rotuladas entre os conceitos. O rótulo pode indicar hierarquia ou influência entre os conceitos.

Atributos: também são conceitos, porém não precisam ser expandidos pois, são conceitos de complexidade mínima no escopo em questão. São representados por nomes ligados aos conceitos através de relacionamentos sem rótulos.

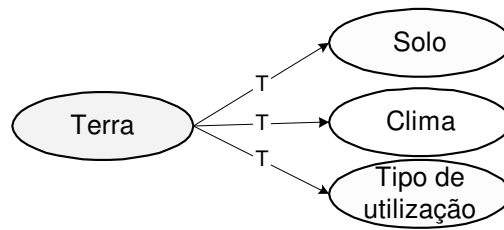


Figura 5: Exemplo, onde os conceitos amarelos são conceitos que serão expandidos e azul o conceito que está em foco, o relacionamento T indica tem.

Essa é a notação geral de um Modelo Conceitual. Porém, pode-se criar outros elementos ou modificar os aqui declarados. Sempre com o intuito de proporcionar maior clareza para os usuários de sistemas ou projetistas. É de suma importância que todas as convenções adotadas sejam especificadas junto a modelagem.

2.7 – Aplicabilidade

Por não se tratar de um modelo lógico, que se detém a representar aspectos de projeto, mas sim um modelo para análise de um certo sistema, o Modelo Conceitual pode ser empregado no desenvolvimento de qualquer tipo de sistema, seja ele computacional ou não. O uso destes modelos se tornam mais e mais indispensáveis à medida que a complexidade do sistema a ser desenvolvido cresce. Projetistas de sistemas de banco de dados, sistemas para gestão, sistemas inteligentes, aplicações para WWW, sistemas de aprendizado, sistemas de informação distribuída e outros, encontram-se preocupados e interessados nesta mudança de concepção da análise de sistemas.

José Palazzo [19], ao citar inúmeras aplicações complexas, principalmente relacionadas ao desenvolvimento de aplicações comerciais na internet, comenta:

“Todas estas aplicações requerem suporte de funcionalidades avançadas e, principalmente, modelos abstratos, metodologias e ferramentas que permitam sua modelagem e projeto”.

3 – Modelagem Orientada a Objetos Através da UML

3.1 – O Modelo Orientado a Objetos

*“A orientação a objetos será a mais importante tecnologia de software emergente da década de 1990.”
Bill Gates, Presidente da Microsoft Corporation. [20]*

A orientação a objetos surgiu com a linguagem Simula em 1967 e só nos anos 80 tornou-se tema central de discussão em programação. Para suprir a necessidade de desenvolver sistemas mais rapidamente a um custo mais baixo, e que embora sejam cada vez mais complexos, precisam ser também mais confiáveis e flexíveis. A Orientação a Objetos deixou de ser apenas um método de implementação, para abranger também o design e a análise.

A tecnologia de objetos promoveu uma mudança na maneira de pensar. A análise O. O. lembra a maneira pela qual os seres humanos naturalmente categorizam e compreendem o mundo (através de conceitos), ou seja, é uma maneira mais natural ao ser humano e mais próxima da realidade. Isto possibilita a construção de sistemas computacionais mais reais e de melhor qualidade. Assim sendo, Rumbaugh define O. O. como “uma nova maneira de pensar os problemas utilizando modelos organizados a partir de conceitos do mundo real. O componente fundamental é o objeto que combina estrutura e comportamento em uma única entidade.” [21]

Enquanto o enfoque do modelo estruturado baseia-se na compreensão do sistema como um conjunto de programas que por sua vez executam processos sobre dados. O enfoque de modelagem por objetos vê o mundo como um conjunto de objetos que interagem entre si. Cada objeto possui atributos ou propriedades que o caracterizam (dados) e operações que representam os processos associados ao objeto no mundo real [21][24].

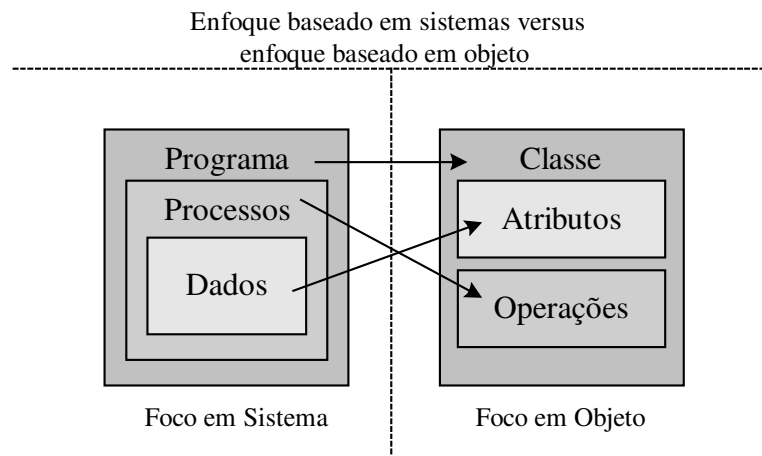


Figura 6: Mudança do enfoque baseado em processos para o baseado em objetos

Esta abordagem significou um grande avanço para o desenvolvimento de sistemas computacionais. Trouxe inúmeros benefícios (ver Tabela 1 [20]) para a indústria de software e simultaneamente para o usuário final.

<i>Reusabilidade</i>	Classes são projetadas para que possam ser reusadas em muitos sistemas, portanto são construídas de forma que possam ser customizadas.
<i>Confiabilidade</i>	Um software construído a partir de classes estáveis provavelmente terá um número menor de falhas.
<i>Integridade</i>	Apenas operações específicas podem acessar certas estruturas de dados.
<i>Programação e Manutenção mais fácil</i>	Devido a modularização os programas são construídos em pequenos módulos, relativamente simples e bem comportados.
<i>Melhor comunicação entre profissionais de informática e pessoas de negócio</i>	Devido a maneira mais natural de ver e tratar o mundo, o modelo O. O. é entendido mais facilmente pelas pessoas de negócio.
<i>Independência do projeto</i>	As classes são projetadas para serem independentes de plataformas, hardware e ambientes de software.
<i>Interoperabilidade</i>	Softwares de fornecedores diferentes podem funcionar juntos e se apresentarem como uma única unidade ao usuário.

Tabela 1:Resumo dos benefícios da tecnologia O. O.

Assim sendo a Orientação a Objetos é de fato a tecnologia para desenvolvimento de sistemas que os usuários (finais ou projetistas) mais almejam em utilizar, o que desencadeou ainda mais o investimento em métodos e metodologias que auxiliem no desenvolvimento de sistemas computacionais baseados nesse paradigma, em todas as fases da engenharia de software. Métodos como o de Booch, OMT, e OOSE entre outros foram criados e ganharam ainda mais força com a unificação – denominada UML.

3.2 – UML (Unified Modeling Language)

A UML surgiu no ano de 1995 com a união dos métodos Booch, OMT e OOSE (Object-Oriented Software Engineering) desenvolvidos respectivamente por Grady Booch, James Rumbaugh e Ivar Jacobson. Com o intuito de prover um método unificado para modelagem de qualquer tipo de sistema computacional.

Além de unir os seus métodos os autores da UML tiveram a preocupação de selecionar e integrar as melhores práticas do mercado, incorporando idéias de vários autores, dentre eles, Peter Coad, Stephen Mellor, Jim Odell, Sally Shlaer, Ed Yourdon, entre outros. A UML recebeu influência das técnicas de Modelagem de Dados (diagramas de entidade e

relacionamento), Modelagem de Negócios (work flow), Modelagem de Objetos e Componentes além de conter novos conceitos que não são encontrados em outros métodos orientados a objetos [21].

Esta unificação tem entusiasmado toda a comunidade de desenvolvedores por, entre tantos motivos, facilitar a comunicação entre os diversos projetistas de um mesmo sistema, já que fornece aos usuários uma linguagem de modelagem visual expressiva (gráfica) e notação padronizada.

Porém não se trata de uma metodologia pois, a UML não prescreve explicitamente um procedimento para sua utilização, podendo, assim, ser utilizada da maneira que se adequar melhor ao gosto do projetista e/ou ao tipo de sistema a ser modelado.

A UML descreve as características dos sistemas através de diagramas O. O. e se propõe a dar suporte às diferentes fases do desenvolvimento de um sistema, desde a análise dos requisitos até a finalização com a fase de testes.

3.3 – Diagramas da UML

A UML é composta por 9 diagramas: de casos de uso, de classes, de objetos, de seqüência, de colaboração, de estados, de atividades, de componentes e de execução. Combinados fornecem uma visão completa do sistema modelado. Os diagramas de classes e de objetos descrevem a estrutura estática do sistema; o comportamento dinâmico é representado pelos diagramas de estado, seqüência, colaboração, e atividade; e o comportamento funcional é suportado pelos diagramas de componentes e execução [23].

Tais diagramas utilizam inúmeros conceitos comuns a O. O. chamados modelos de elementos, como por exemplo classes, estados, relacionamentos, e provêm também o uso de mecanismos gerais que funcionam como comentários, informações adicionais, além de mecanismos de extensão para adaptar ou estender a UML para um usuário específico.

Estes elementos não são descritos neste relatório, já que a notação utilizada pela UML nos diagramas não é o foco deste trabalho, mas podem ser encontrados em detalhes em [21][23][25][26][27][28][29][30] e no Apêndice A.

3.3.1 – Diagrama de Casos de Uso

Este diagrama é utilizado para capturar a funcionalidade do sistema. Ele é descrito em termos de Atores que são as entidades externas, de casos de uso que representam cada uma das funcionalidades do sistema e, relacionamentos entre os casos de uso e os atores.

É recomendado que o diagrama de casos de uso esteja acompanhado de uma especificação contendo informações sobre cada relacionamento, incluindo nome do caso de uso, quem inicia a ação e sua descrição.

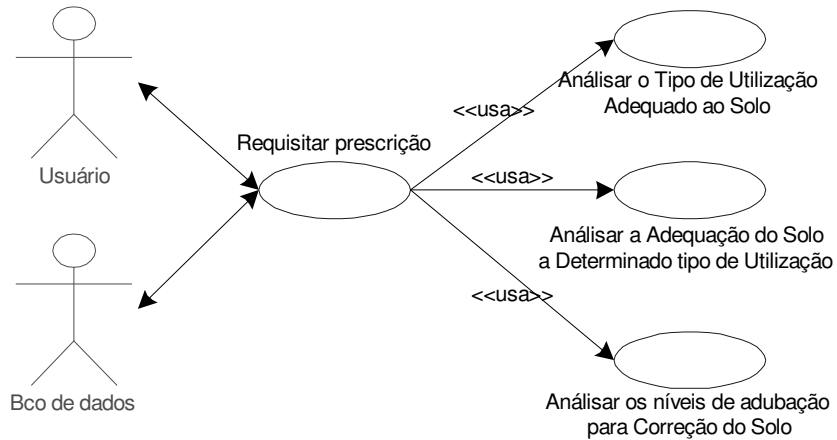


Figura 7: Diagrama de Caso de Uso

3.3.2 – Diagrama de Classes

O diagrama de classes descreve o conhecimento envolvido no problema, demonstrando a estrutura estática do sistema. Este diagrama apresenta as várias classes com seus conteúdos (atributos e operações) e relacionamentos entre elas.

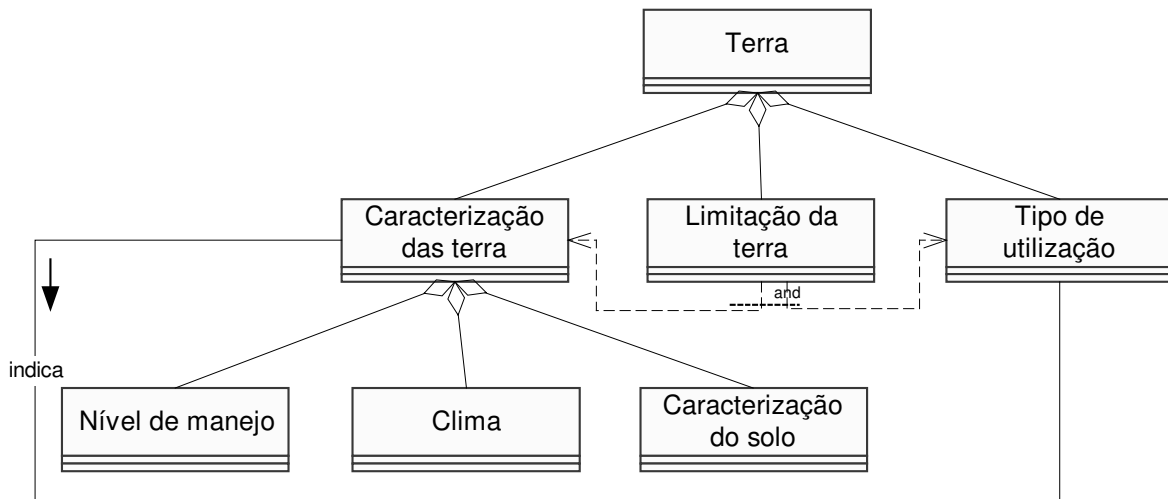


Figura 8: Diagrama de Classes

3.3.3 – Diagrama de Objetos

O diagrama de objetos representa uma instanciação do diagrama de classes. Este diagrama não é de importância fundamental, ele serve basicamente para exemplificar os diagramas de classes.

3.3.4 – Diagrama de Interação

Este diagrama descreve a interação dinâmica entre os vários objetos de um sistema. Ele serve principalmente para captar a seqüência de mensagens enviadas entre os objetos. É geralmente utilizado para mostrar o funcionamento de casos de uso em termos de suas interações entre classes. São compostos pelos diagramas de seqüência e de colaboração.

3.3.4.1 – Diagrama de Seqüência

Este diagrama mostra interações de objetos organizados em uma seqüência de tempo e de mensagens trocadas. Cada objeto possui uma linha de vida e a seqüência de interações são representadas através de setas entre os objetos que se relacionam.

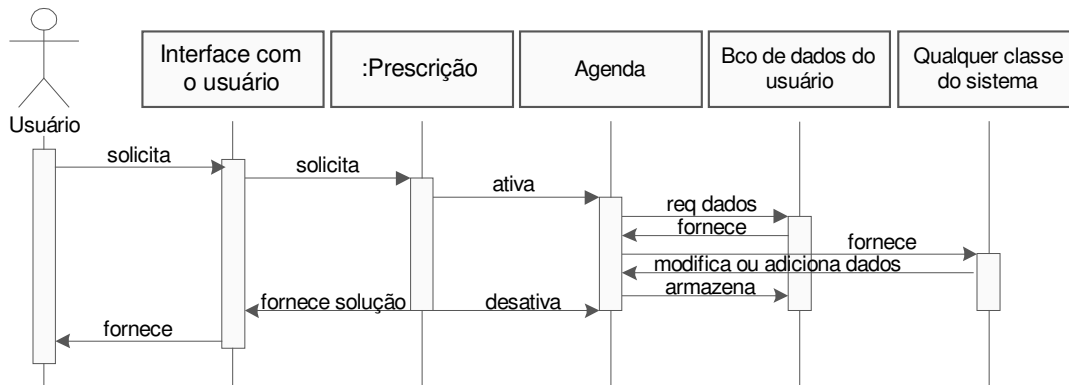


Figura 9: Diagrama de Seqüência

3.3.4.2 – Diagrama de Colaboração

O diagrama de colaboração mostra o contexto completo de uma interação, ou seja, além da troca de mensagens mostra também os relacionamentos entre objetos. Este diagrama é projetado como um diagrama de objetos que são apresentados juntamente com seus relacionamentos.

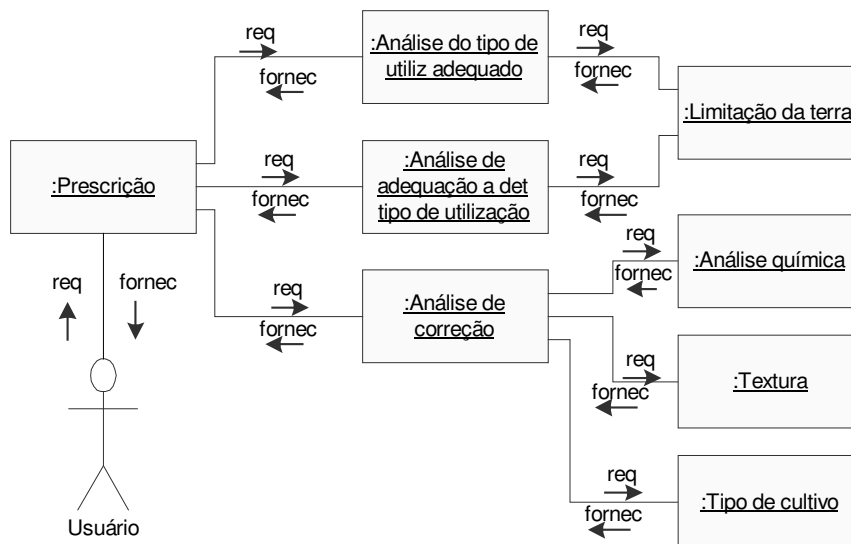


Figura 10: Diagrama de Colaboração

3.3.5 – Diagrama de Estados

Este diagrama serve como um complemento para a descrição das classes. Ele relaciona os possíveis estados que os objetos das classes podem ter e quais eventos podem causar a mudança de estado [23]. Como captura todos os possíveis estados de um sistema, este diagrama deve ser construído para cada classe individualmente, para que um grande número de estados não dificulte o estudo e entendimento do sistema.

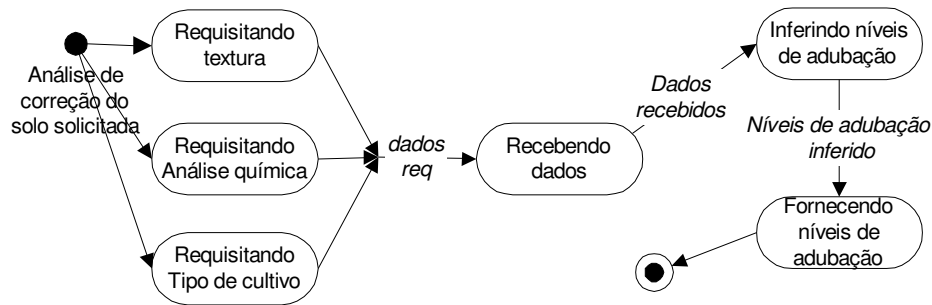


Figura 11: Diagrama de Estados

3.3.6 – Diagrama de Atividades

É um caso especial de diagrama de estado, onde os estados denotam ações e as transições são ativadas pela conclusão de ações nos estados precedentes [21]. Este diagrama pode capturar basicamente o funcionamento interno de um objeto, ou seja, uma dinâmica intrínseca; as ações que serão realizadas quando uma operação é executada; como um grupo de ações relacionadas podem ser executadas e como elas vão afetar os objetos em torno delas; ajudam na compreensão do fluxo de trabalho entre casos de uso.

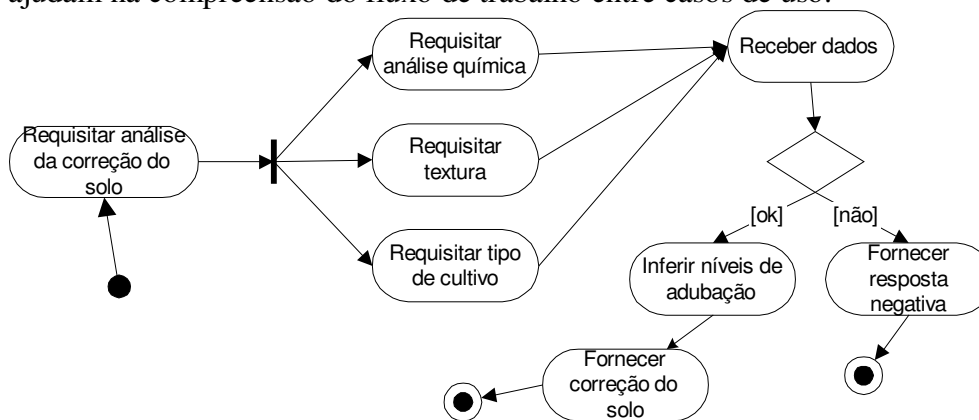


Figura 12: Diagrama de Atividades

3.3.7 – Diagrama de Implementação

Ilustra a arquitetura física em termos de hardware e software que cercam a implementação do sistema. Consiste na descrição dos componentes e módulos (nós) incluindo

dependências e associações entre eles. Os diagramas de Implementação podem ser de Componentes e de Execução.

3.3.7.1 – Diagrama de Componentes

Descreve os componentes de software e suas dependências. Eles são basicamente os arquivos implementados no ambiente de desenvolvimento.

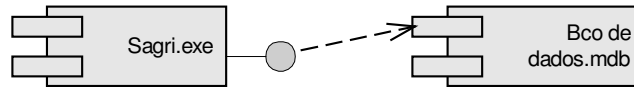


Figura 13: Diagrama de Componente

3.3.7.2 – Diagrama de Execução

O diagrama de execução ou de implantação descreve a arquitetura física do hardware e do software no sistema. É composto por nós que são os objetos físicos como “máquina servidora”, “impressora” etc, componentes e conexões entre nós e componentes.

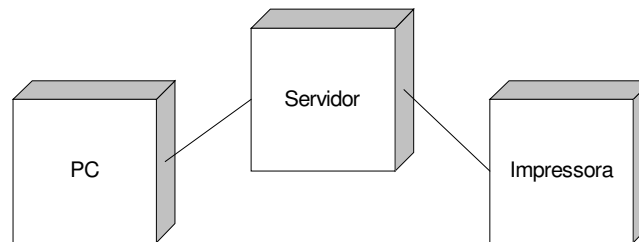


Figura 14: Diagrama de Execução

3.4 – Utilização da UML

*“A UML é uma linguagem de modelagem não é uma metodologia.”
- José Davi Furlan, 1998 – [21]*

A UML especifica uma linguagem para modelagem mas não descreve explicitamente um procedimento de utilização indicando uma ordem de atividades que devem ser realizadas para se alcançar o objetivo final que é o de desenvolvimento de sistemas. Há portanto a necessidade de se adotar um processo para a utilização da UML.

3.4.1 – Fases do Desenvolvimento de um Sistema Computacional

Um dos processos tradicionais de desenvolvimento O. O. é dividido nas 5 fases: análise de requisitos, análise, projeto, implementação e testes, às quais deve-se aplicar um ciclo de vida em espiral de forma que se tenha sempre um modelo mais completo do sistema a cada volta na espiral, permitindo que se aplique a abordagem de prototipação em qualquer etapa da evolução do sistema [7].

3.4.1.1 – Análise de Requisitos

O objetivo desta fase é descrever o que o sistema deve fazer. Quais as intenções e necessidades dos usuários, ou seja, a análise de requisitos apresentará toda a funcionalidade que o sistema deverá possuir, sem se preocupar em como ela será implementada.

3.4.1.2 – Análise

A meta da Análise é prover uma compreensão do sistema descrevendo o conhecimento do domínio e facilitar a comunicação entre os técnicos e usuários, não detendo-se a soluções técnicas ou detalhes de código ou programas.

3.4.1.3 – Projeto

A fase de Projeto expande os modelos obtidos da Análise em soluções técnicas, descreve a visão interna e o funcionamento das requisições do usuário. O Design ou Projeto resulta no detalhamento das especificações para a fase de Implementação do sistema [23].

3.4.1.4 – Implementação

Na Implementação os modelos obtidos das outras fases são transformados em código. A tarefa de codificação será facilitada se o Projeto tiver sido elaborado corretamente e com detalhes suficientes. Nesta fase verifica-se ainda a necessidade da criação ou modificação de operações que resulta em alteração dos modelos do Projeto.

3.4.1.5 – Testes

Nesta fase avalia-se o sistema através de diferentes testes em busca de erros. Primeiramente testando cada um dos requisitos individualmente e depois testando o sistema como um todo.

Na Tabela 2 são apresentadas as fases de desenvolvimento com os respectivos diagramas adequados a cada fase.

Fases	Diagramas Utilizados
Análise de Requisitos	De Casos de uso
Análise	De Casos de uso, de Classes, de Sequência, de Estado
Projeto	De Classes, de Colaboração, de Atividades, de Componentes e de Execução
Implementação	Alteração dos diagramas construídos anteriormente.
Testes	Alteração dos diagramas construídos anteriormente.

Tabela 2: Processo Tradicional de desenvolvimento O. O. e diagramas UML adequados a cada fase

A UML proporcionou vários benefícios a comunidade de desenvolvedores, já que padronizou uma linguagem gráfica para ser utilizada pelos projetistas, e induziu a criação de ferramentas CASE baseadas nesta linguagem, além de fornecer os benefícios oferecidos pela

Orientação a Objetos, como por exemplo, melhor comunicação entre profissionais de informática e pessoas de negócio. Oferece ainda, liberdade para ser utilizada da maneira que mais se adequar ao desenvolvimento do sistema. Porém esta mesma liberdade pode fazer com que se utilize uma metodologia inadequada e assim levar à construção de um modelo que não abranja todas as visões necessárias para o desenvolvimento do sistema, não permitindo que o projetista parta direto da modelagem para a implementação ou mesmo que o projetista consuma muito tempo desenhando diagramas muito detalhados, que não seriam necessários.

A UML é uma técnica de ampla aplicabilidade, é adequada a descrição de qualquer sistema computacional, com base no paradigma Orientado a Objetos, desde que seja adotada uma metodologia adequada ao desenvolvimento do sistema, e principalmente que já se tenha realizado um bom levantamento do conhecimento intrínseco ao sistema e se conheça bem a solução para o problema proposto.

4 – Modelagem do Sistema

4.1 – O Sistema SAGRI

O sistema SAGRI - Sistema Inteligente de Apoio a Produção Agrícola - é um sistema que se destina a auxiliar agricultores em suas tarefas no campo, como por exemplo no plantio, correção de solos, adubação etc, de forma a maximizar a produtividade e o lucro. Trata-se de um sistema amplo de muita complexidade, que envolve uma grande quantidade de tecnologias como por exemplo, banco de dados, base de conhecimento, SIG, redes neurais, lógica fuzzy etc, para que seja efetivamente concebido e implementado.

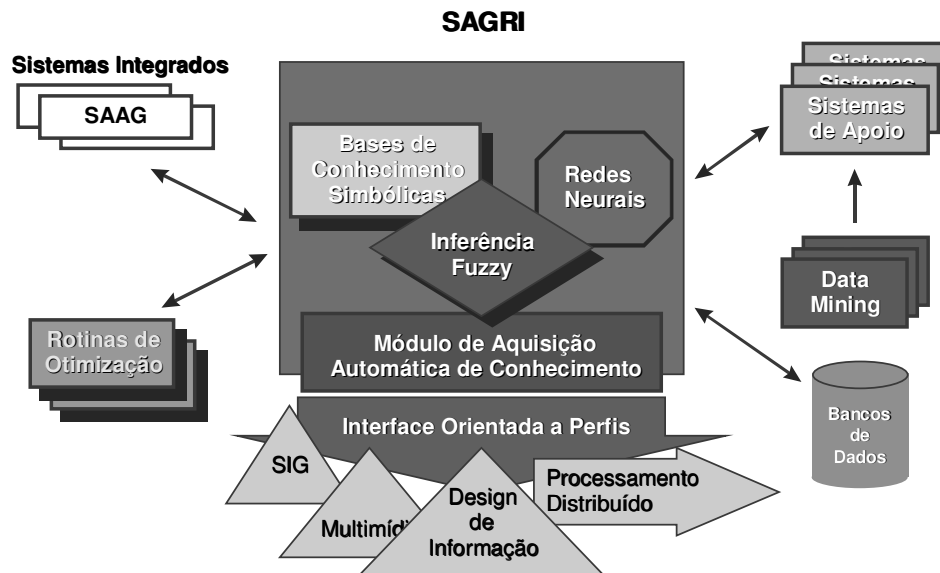


Figura 15: Estrutura conceitual do Sagri

O SAGRI é caracterizado principalmente por suas Bases de Conhecimento. Estas Bases detêm todo o conhecimento relacionado a agricultura, obtido da representação do conhecimento feita na fase de Aquisição vinda a partir de especialistas, livros, informativos e outras fontes.

Esta é a parte mais complexa do desenvolvimento do sistema, já que as informações sobre agricultura são na maioria das vezes incertas, incompletas ou mesmo contraditórias, exigindo dos analistas de sistemas um maior cuidado, e uma melhor análise do sistema, desde a aquisição até a própria implementação.

A aquisição é uma das fases mais importantes para o projeto: Primeiro, por representar a base de todo o sistema, é nela que se obtém todo o conhecimento, informações e dados para o sistema. É necessário, portanto, haver cuidados como: conduzir entrevistas ou estudos de forma que se adquira uma grande quantidade de informações, organizadamente pois, não basta deixar o entrevistado a vontade para transmitir as informações, é necessário também que este comunique-as de maneira “significativa” para o entrevistador; documentar exatamente o que o especialista informou, sem dar margem a ambigüidades; E segundo, por fazer uma “pré-

codificação” do conhecimento adquirido em uma linguagem simples, entendível por todos os participantes do projeto, incluindo nesta codificação as informações necessárias ao sistema, eliminando discussões ou outros fatores irrelevantes para o sistema, que ocorreram durante a aquisição.

Há portanto, uma grande preocupação em se fazer uma representação deste conhecimento de maneira clara e precisa, sem que se perca uma única parcela das informações, para que mais tarde o sistema ou mesmo o projeto não sofra danos como incredibilidade dos usuários ou grandes modificações por parte dos projetistas.

Outra grande dificuldade encontradas nesta fase foram: a de compreender as diferentes linguagens que os especialistas utilizam (há uma linguagem mais científica utilizada por técnicos e engenheiros agrônomos, uma linguagem mais coloquial utilizada pelos agricultores e uma linguagem intermediária entre os dois extremos); tratar as diferentes visões dos especialistas pois, estes se baseiam em suas experiências, sem haver nada formalizado, e que muitas vezes são contraditórias; manipulação de conceitos imprecisos, comumente usados pelos especialistas desta área como, por exemplo, “muito”, “pouco”, “influencia”, “bom”, “ruim” etc, sem no entanto, haver uma quantização fixa para cada um deles pois, dependem da região ou mesmo da visão do especialista; dificuldades como falta de dados e dados atualizados sobre a nossa região também foram encontradas.

No entanto todas estas dificuldades, existentes a milhares de anos, não são suficientes para impedir que o homem deixe de investir na agricultura. Este investimento é ainda hoje, a base da economia de muitos países, e põe a mesa da maioria de brasileiros pois, a inteligência humana não é limitada por imprecisão, incerteza e nem mesmo inconsistência. E é isto o que se espera de sistemas computacionais. Que eles sejam hábeis o suficiente para tratar este tipo de conhecimento. Capazes de fornecer ao usuário uma solução adequada ao problema ou pelo menos uma resposta, justificando sua incapacidade de resolução daquele problema.

No entanto, é necessário muito mais que tecnologias aplicáveis ao tratamento deste tipo de conhecimento. É indispensável um mapeamento claro e preciso, um estudo bem elaborado do sistema real para se ter uma compreensão mais ampla do assunto em questão e só então analisar qual tecnologia se adequará mais a cada parte do sistema.

Em momento algum deve-se esquecer que o objetivo de qualquer sistema computacional é oferecer uma solução plausível para um problema do mundo real, e não, simplesmente aplicar métodos computacionais a problemas do mundo real. Portanto deve-se ter um amplo domínio sobre os conceitos e relacionamentos intrínsecos ao problema do mundo real, sejam eles exatos ou inexatos, completos ou incompletos, precisos ou imprecisos, consistentes ou inconsistentes, sem, no entanto, desejar tornar este conhecimento adequado aos métodos computacionais pois, o que se ambiciona não é moldar o mundo mas sim modelá-lo de forma a obter um maior conhecimento sobre ele, e então, construir ferramentas que se adequem à aplicação.

A agricultura é uma área que ainda hoje, pelo menos aqui no Brasil, não aplica intensivamente a informática no processo de solução de suas atividades, com certeza pela limitação financeira, mas também, por falta de credibilidade na tecnologia ou por pensar que o conhecimento da área é intratável por “máquinas” ou ainda por achar que é impossível uma “máquina” possuir conhecimento e conseguir manipulá-lo (Raciocinar, ter inteligência).

Para tratar a incredibilidade das pessoas desta área, o SAGRI tem que ser capaz de mostrar o seu raciocínio na resolução de um dado problema, mostrando como conseguiu chegar a tal conclusão, assim como se apresentar de uma maneira fácil de compreender, agradável de ver e, principalmente, dar respostas bem elaboradas e mais inteligentes, mesmo que o usuário não tenha todas as informações necessárias para resolução de seu problema.

Este é um outro fator de extrema importância pois, o agricultor nem sempre tem disponível todas as informações necessárias ao processamento de uma certa tarefa (por questões financeiras ou pela própria falta de conhecimento), como por exemplo a composição química do solo de uma determinada gleba de sua propriedade.

Mas mesmo assim é possível atingir o objetivo por outros caminhos, mesmo que estes não sejam os mais adequados. Para isto é necessário oferecer ao sistema computacional uma grande quantidade de conhecimento bem estruturado e organizado, e oferecer em conjunto a possibilidade de manipulação deste conhecimento. Temos que ter em mente que a máquina ainda não é capaz de estruturar significativamente o conhecimento, esta é a parte mais difícil e complexa e cabe a nós, seres humanos.

Para atingir seu objetivo, este sistema tem que ser capaz de tratar as inúmeras problemáticas da produção agrícola, e portanto ser capaz de tratar todo o conhecimento da área, tal como ele é, o mais condizente possível com a realidade, retratando o mundo e o próprio raciocínio humano sobre este mundo.

Além das características já citadas, há a necessidade de representar e manipular o conhecimento de diversas áreas e domínios para que se tenha uma conclusão efetiva, que represente uma informação utilizável pelos agricultores e/ou agrônomos. Assim, é de grande importância possuir um modelo conceitual bem elaborado.

Para lidar com as discrepâncias do mundo real e oferecer ao projetista subsídios indispensáveis ao processo de desenvolvimento do sistema, de forma a torná-lo um sistema que realmente atende as necessidades do usuário, foi realizada uma modelagem conceitual bastante ampla para o Sagri, capaz de fornecer conhecimento para inúmeros problemas através de caminhos diferentes, como por exemplo, é possível encontrar solução para a indicação ao usuário de qual a “textura” de seu solo através de sua “análise granulométrica”, ou se o usuário não possuir esta análise, o sistema deve ser capaz de estimar a resposta através da “sensação ao tato”; lembrando que, na maioria das vezes, só através da modelagem é possível encontrar diferentes caminhos de se atingir um mesmo objetivo pois, o conhecimento adquirido de especialistas geralmente não é transmitido de forma estruturada o suficiente para identificá-los facilmente.

Uma outra preocupação na fase de modelagem do conhecimento do SAGRI foi em ser sempre suficientemente compreensível pelos outros projetistas do sistema, servindo como meio de propagação das informações contidas nela, bem como, mostrar aos especialistas participantes que o sistema está sendo construído a partir de informações entendidas da maneira correta, informações verdadeiras.

Um outro papel que os modelos do SAGRI desempenham é o de esclarecer dúvidas não só durante a aquisição, mas também depois, durante a fase de representação e implementação do conhecimento. Dúvidas estas que vão desde o entendimento de algum

conceito, até à veracidade, precisão e completude dos dados que estão sendo fornecidos pelos vários especialistas, que naturalmente possuem visões diferentes.

4.2 – Descrição do Problema

O Sagri tem o objetivo de fazer uma análise da terra, fornecendo ao usuário uma prescrição. Esta prescrição deve ser constituída do tipo de utilização adequado ao solo da propriedade ou se o tipo de utilização que se pretende desenvolver nas terras é realmente adequado ao solo e, também, fornecer uma sugestão de níveis de adubação dada uma certa cultura.

Cada usuário do Sagri deverá ter cadastrado no Banco de dados uma Propriedade, particionada ou não, em Glebas. O sistema interage com este Banco de Dados requisitando as informações relacionadas à Propriedade e também fornecendo informações para serem armazenadas.

As características do solo são encontradas a partir de uma inferência realizada sobre o conhecimento de solos adquirido e sobre as informações fornecidas pelo usuário. A solução do problema foi projetada de maneira que o usuário precise fornecer o mínimo de informações possível pois, se tratando da área agrícola, os conceitos necessários são em sua maioria, conceitos não identificados facilmente pelo agricultor sendo necessário detectá-los através de informações que o agricultor possa de fato fornecer. Como por exemplo, o conceito de “aeração”, o agricultor não é capaz de detectá-lo, e às vezes, nem mesmo de compreendê-lo mas pode fornecer dados de um outro conceito como “textura” que lhe é familiar e de possível identificação, a partir do qual o sistema irá inferir a “aeração”.

Dada a descrição geral do problema, será apresentada a modelagem conceitual e a modelagem Orientada a Objetos através da UML construídas durante o desenvolvimento deste trabalho.

4.3 – Modelagem Conceitual

A modelagem conceitual do Sagri foi iniciada a partir do modelo estático, visto que não se conhecia ainda um processo de solução para que o objetivo fosse atingido.

No decorrer da aquisição de conhecimento é que foi se identificando os conceitos importantes e os relacionamentos entre eles, mapeando as informações à luz dos objetivos propostos. Identificar o conhecimento que seria realmente necessário sem saber como seriam manipulados foi tarefa difícil, levando muitas vezes a um mapeamento de conceitos dispensáveis ao processo de solução escolhido. No entanto, isto não chegou a ser um desperdício pois, o tempo e o trabalho consumidos com estes conceitos não foi tão extenso mas proporcionou ao modelo conhecimento suficiente para ser utilizado por outros processos de solução.

O modelo gerado é flexível e rico em detalhamento, servindo não somente aos propósitos hora especificados ao Sagri mas aos profissionais da área, tanto para ensino como subsidiando pesquisas. Os conceitos estão descritos e o relacionamento entre eles mapeados.

O processo de solução desenvolvido foi baseado em [31], escolhido por ter sido o único método realmente documentado e utilizado encontrado pela equipe de desenvolvedores do sistema, embora o Modelo Conceitual construído permita inferir outros processos de solução.

A seguir é apresentada uma parte da modelagem do sistema Sagri, a modelagem completa está exposta no Apêndice B. Nestes modelos foram acrescentados alguns elementos em sua notação que são descritos no decorrer de sua exposição.

Ambas as modelagens, estática e dinâmica, serão apresentadas seguindo uma seqüência didática. Será apresentado, primeiramente, o modelo dinâmico e depois, o modelo estático, sendo que foram construídos paralelamente e só após uma grande interação do modelador com os modelos é que se obteve o Modelo Conceitual que aqui é apresentado.

4.3.1 – Definição do Problema

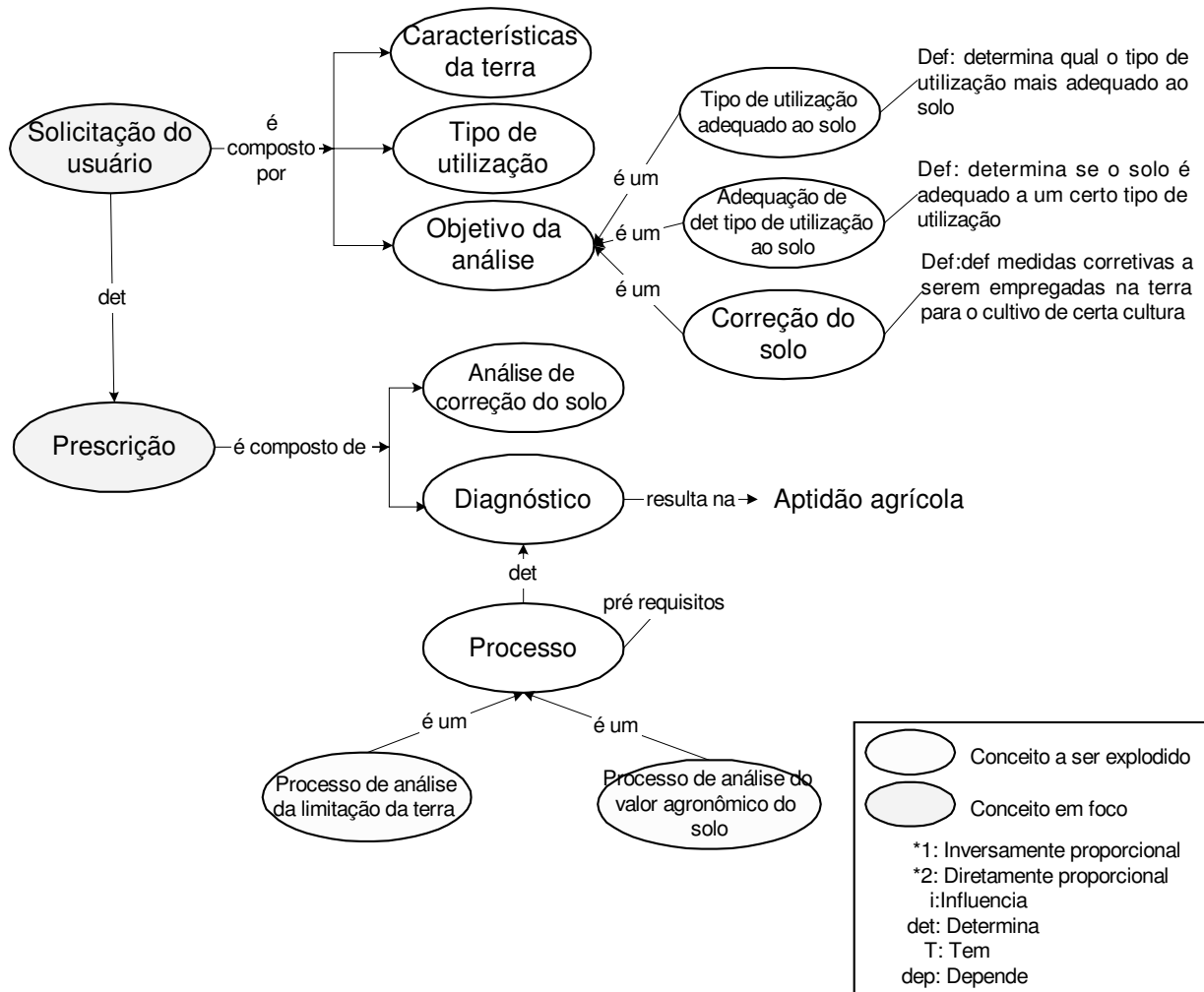


Figura 16: Definição do problema

Como foi dito, o objetivo do sistema é fazer uma análise da terra, fornecendo ao usuário uma prescrição contendo a aptidão agrícola do solo e uma análise de correção. Este diagnóstico será obtido através de algum processo de solução, neste caso foram identificados dois processos: o baseado nas limitações da terra e o baseado no valor agrônômico do solo. O processo que será utilizado no decorrer do estudo de caso é baseado nas limitações da terra. O outro processo será descrito superficialmente, no contexto deste trabalho.

4.3.2 – O Modelo Dinâmico

O modelo dinâmico, aqui desenhado, é o *Processo de análise da limitação da terra*. Este processo de solução baseia-se nas limitações oferecidas pela terra como meio para se encontrar a aptidão agrícola.

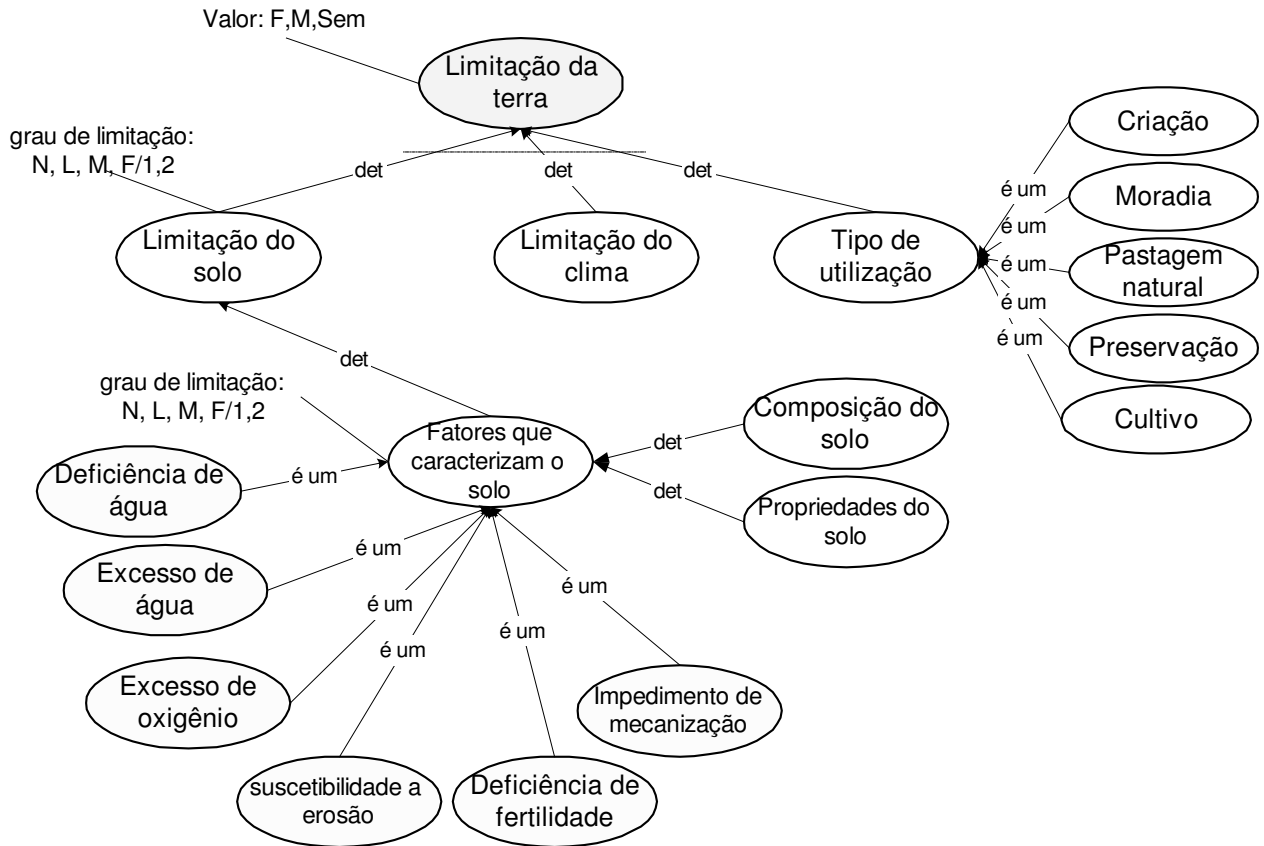


Figura 17: Limitação da terra

Para determinação da limitação da terra, é necessário determinar a *Limitação do solo*, a *Limitação do clima* e o *Tipo de utilização*. Por sua vez a *Limitação do solo* é determinada pelos *Fatores que caracterizam o solo*: a *Deficiência de água*, o *excesso de água*, o *excesso de oxigênio*, a *suscetibilidade à erosão*, a *deficiência de fertilidade*, e o *impedimento à mecanização*, todos determinados pela *Composição do solo* e *Propriedades do solo*. Expandindo cada um destes *Fatores que caracterizam o solo* temos:

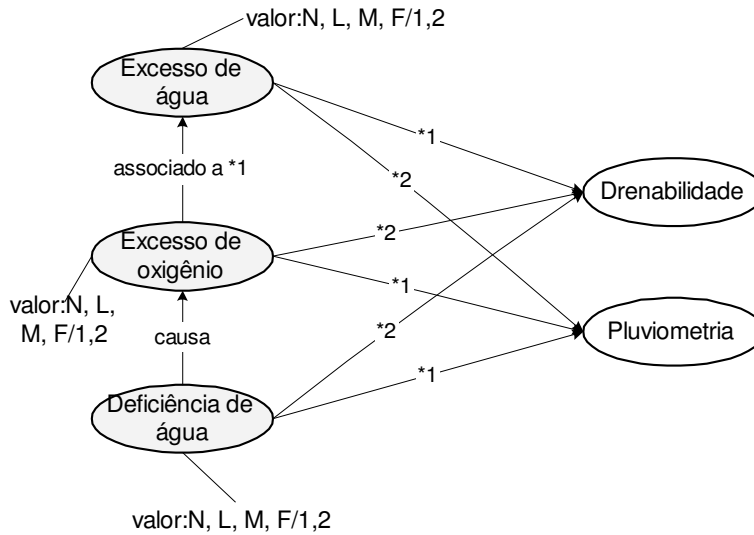


Figura 18: Excesso de água, Excesso de oxigênio e Deficiência de água

A figura 18 mostra que o *Excesso de água*, o *Excesso de oxigênio* e a *Deficiência de água* estão associados a *Drenabilidade* e a *Pluviometria*, e também que há uma associação entre estes fatores.

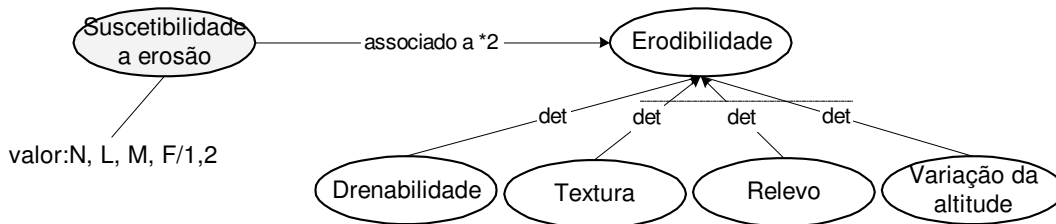


Figura 19: Suscetibilidade à erosão

A *Suscetibilidade à erosão* está associada diretamente a *Erodibilidade*. Que por sua vez é definida pela *Textura*, *Relevô*, *Variação de Altitude* e *Drenabilidade*.

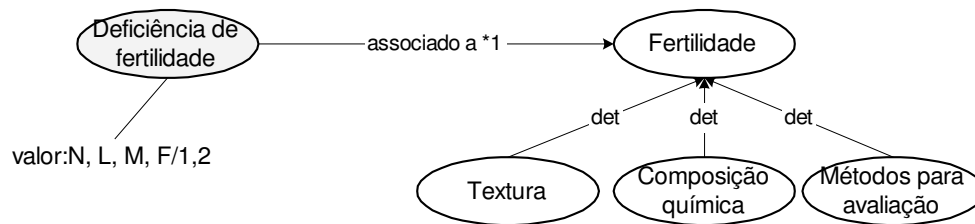


Figura 20: Deficiência de fertilidade

A *Deficiência de fertilidade* está associada inversamente a *Fertilidade*, que por sua vez pode ser determinada pela *Textura*, *Composição química*, e *Métodos para avaliação*.

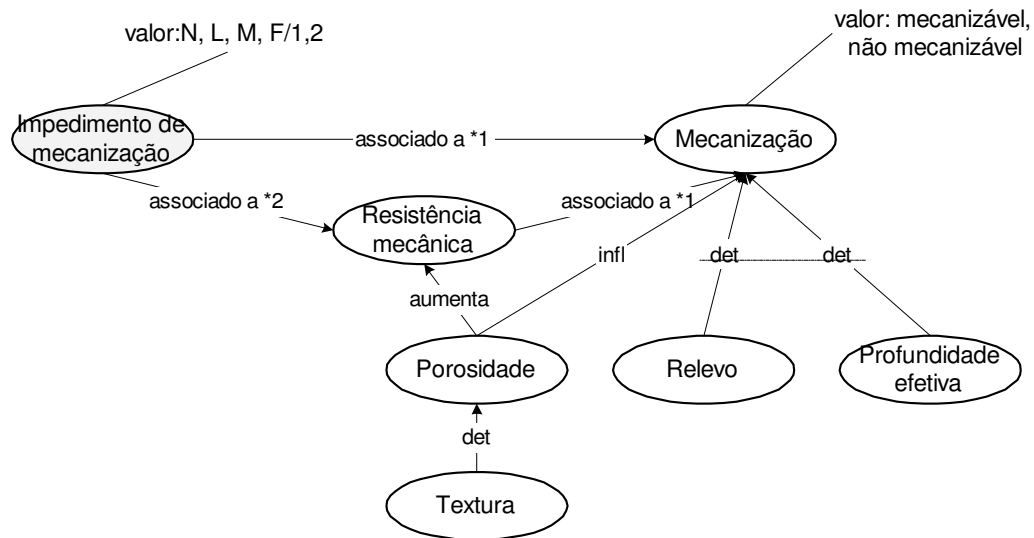


Figura 21: Impedimento de mecanização

O *Impedimento de mecanização* está associado inversamente a *Mecanização* e diretamente a *Resistência mecânica*, que por sua vez podem ser determinados por outros conceitos.

Paralelamente são expandidos cada um dos conceitos citados até agora, como *Textura*, *Drenabilidade*, *Fertilidade* etc. constituindo a modelagem estática do Modelo Conceitual. É importante ressaltar que não existe uma fronteira bem delimitada entre o modelo estático e o modelo dinâmico, eles se mesclam para descrever o mundo real e o processo de solução proposto, constituindo o Modelo Conceitual do sistema.

4.3.3 – O Modelo Estático

A modelagem estática é constituída pela definição dos conceitos e associações entre eles, fornecendo subsídios para outros modelos dinâmicos.

O conceito *Terra*

Terra é um conceito bastante utilizado pelo agricultor e define o contexto ambiental, caracterizado pelo clima e pelo solo.

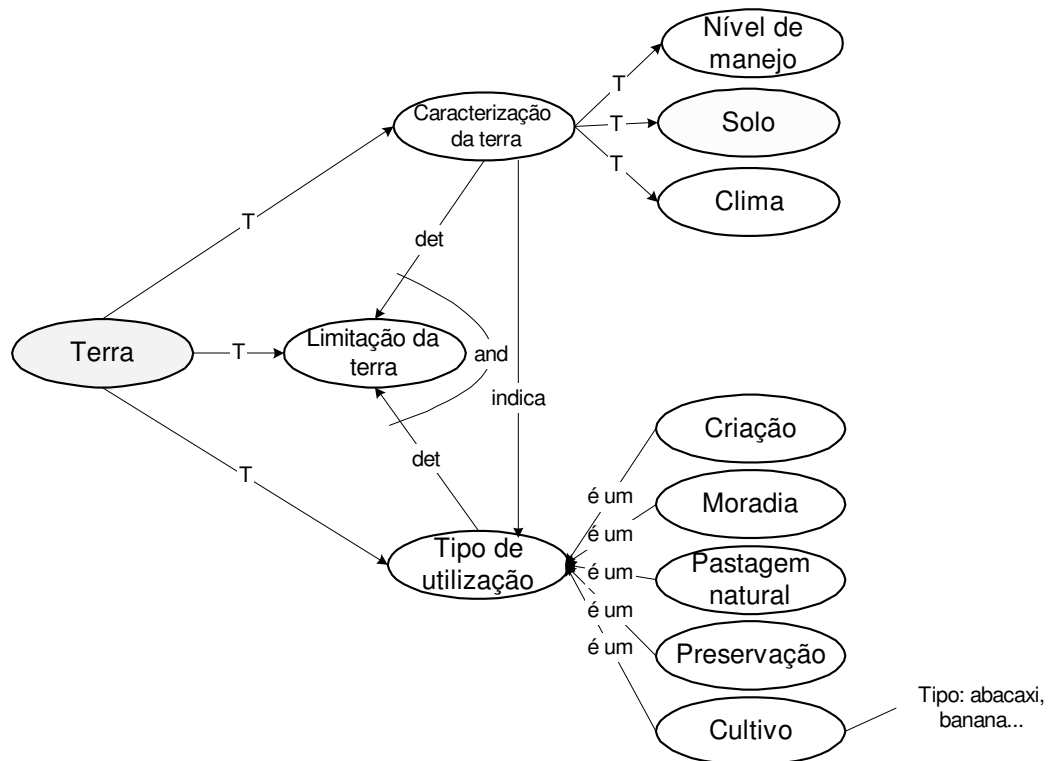


Figura 22: O conceito Terra

O conceito *Solo*

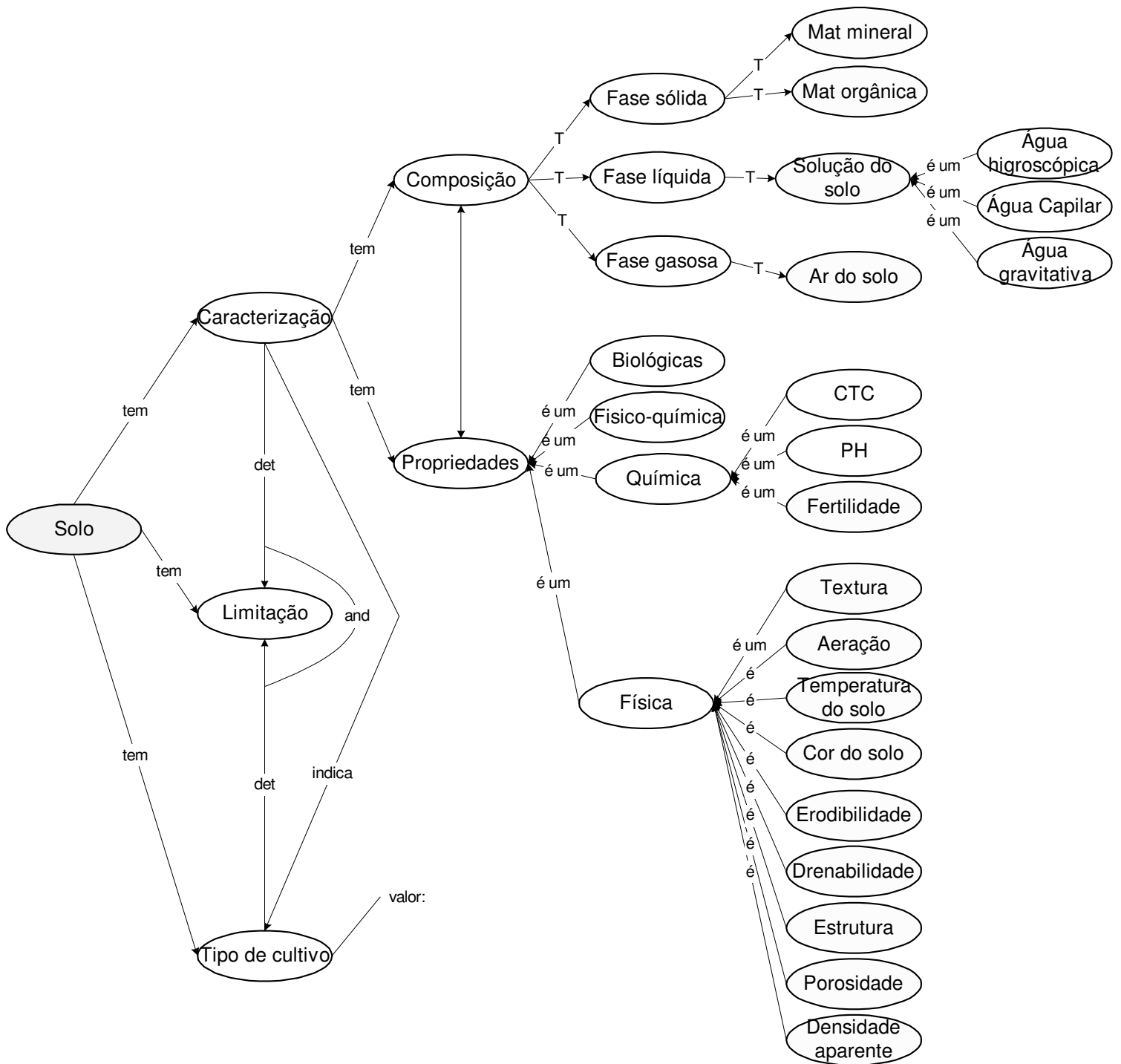


Figura 23: O conceito Solo

O conceito Solo representa, neste trabalho, o conceito principal. Toda a aquisição de conhecimento realizada teve como objetivo identificar as características e influências dos conceitos relacionados ao solo, a fim de determinar a aptidão agrícola.

A partir daqui se expandirão todos os conceitos até se obter uma boa descrição conceitual dos aspectos do solo à luz do objetivo e escopo delimitados ao definir o problema como, por exemplo, o conceito Textura.

Textura e Matéria mineral

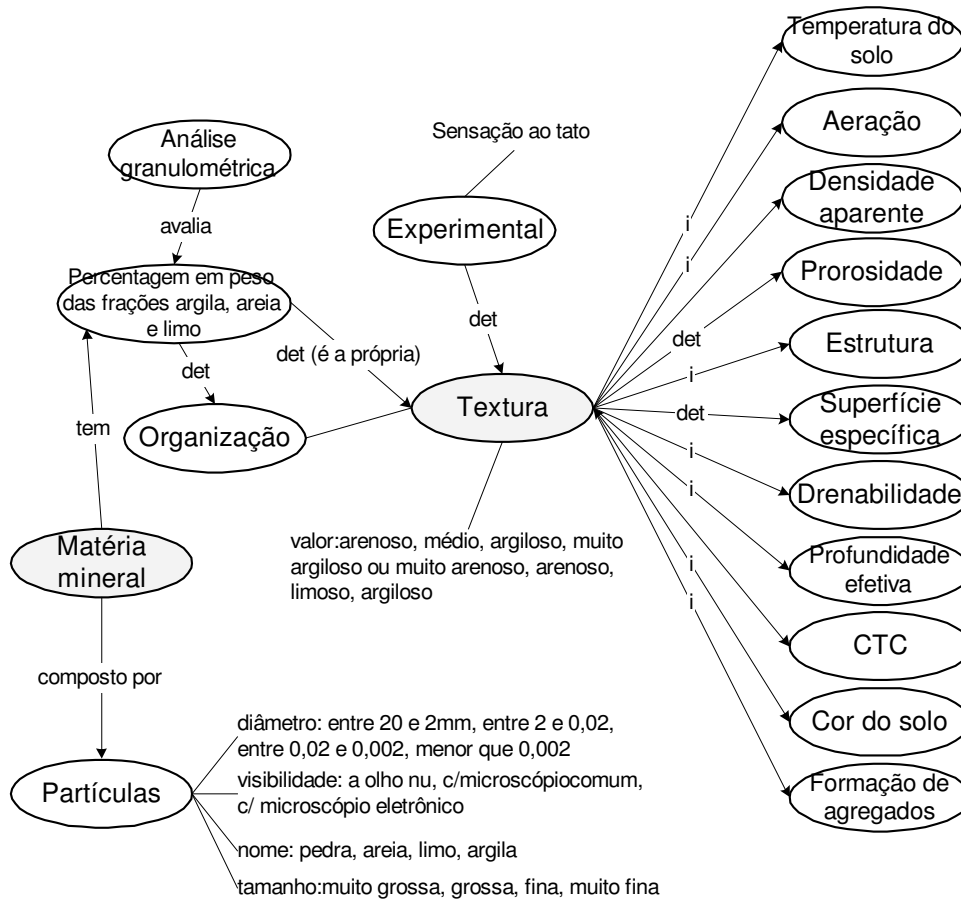


Figura 24: O conceito Textura e Matéria mineral

O conceito Textura é uma das propriedades do solo, a partir dele pode-se ter uma indicação de várias outras propriedades como Temperatura do Solo, Aeração, etc (ver figura 24).

A expansão dos outros conceitos podem ser encontrados no Apêndice B.

4.4 – Modelagem Orientada a Objetos Através da UML

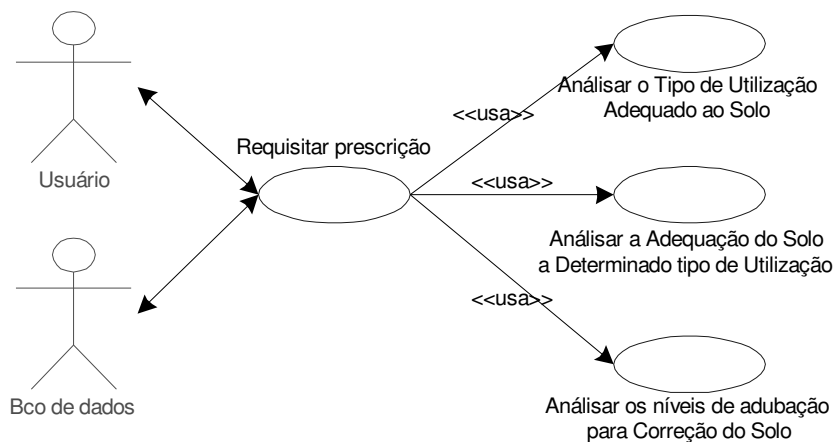
Esta modelagem teve como base os modelos conceituais feitos durante a aquisição de conhecimento, posterior análise da corretude dos modelos desenhados e estudo de um processo de soluções adequado.

4.4.1 - Diagrama de Casos de Uso

O primeiro passo para análise é definir os casos de uso que descrevem o quê o sistema oferece em termos de funcionalidade. Os atores envolvidos neste sistema são os usuários, que podem ser agricultores, agrônomos ou qualquer pessoa que se interesse em saber qual a aptidão agrícola do solo que possui. E também o Banco de dados do sistema.

O usuário se comunica com o caso de uso “Requisitar prescrição” em busca de um diagnóstico para sua terra. Este caso de uso utiliza os três outros casos de uso de acordo com o objetivo do usuário e interage com o Banco de dados, em busca de informações ou para armazenar novas informações relativas a Propriedade do usuário.

O diagrama de caso de uso previsto para o sistema Sagri é:



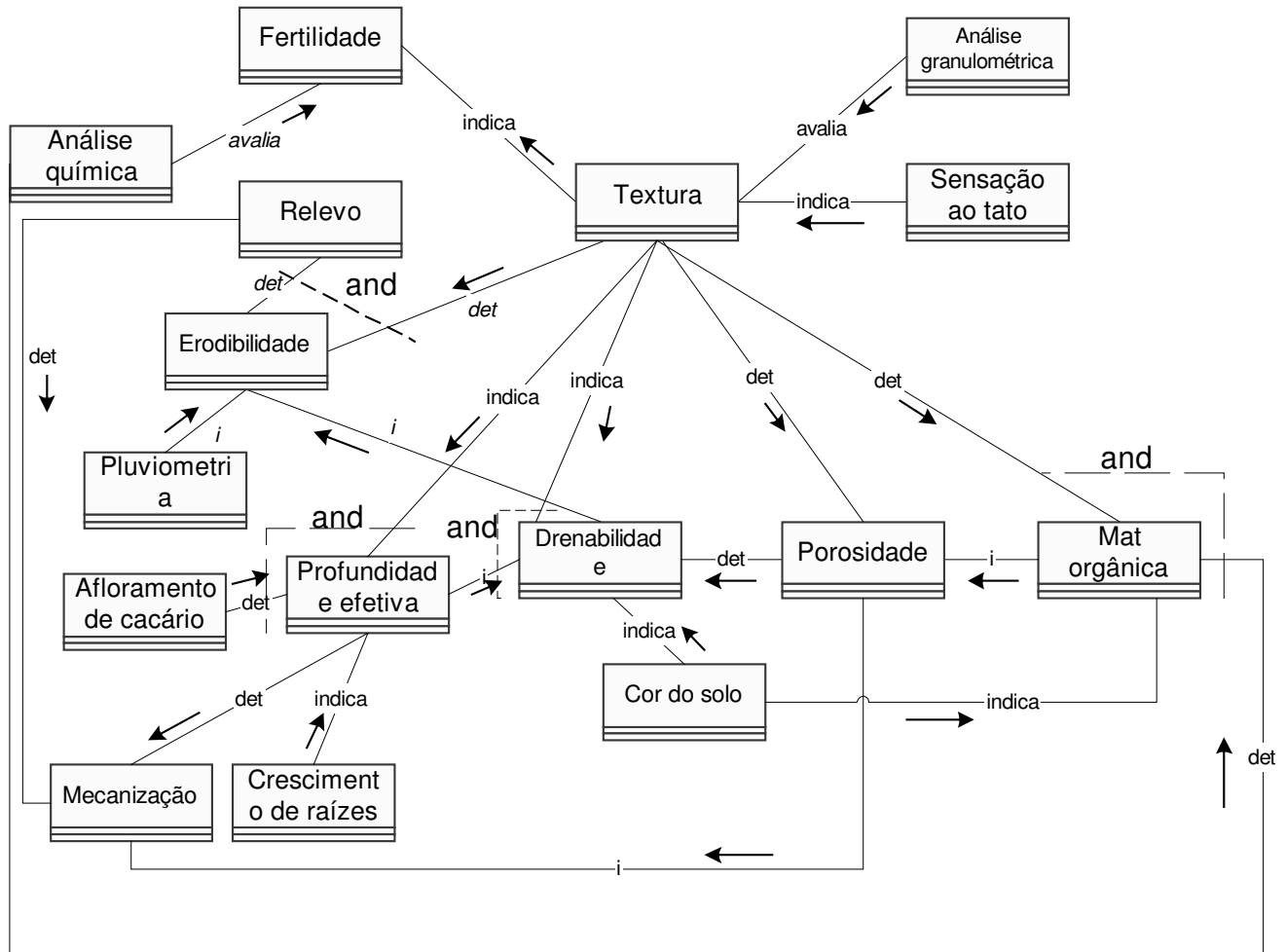
<i>Nome do caso de uso</i>	<i>Quem inicia o caso de uso</i>	<i>Descrição do caso de uso</i>
Requisitar prescrição	Usuário	O usuário deverá informar o objetivo da consulta. O sistema retornará uma prescrição do solo baseado nos casos de usos posteriores. Interage também com o Banco de dados, obtendo e armazenando informações.
Analisar o tipo de utilização adequado ao solo	Prescrição	Solicita ao usuário algumas informações e retorna uma utilização adequada ao solo.
Analisar a adequação do solo a determinado tipo de utilização	Prescrição	Solicita ao usuário algumas informações e retorna a adequação de uma determinada cultura ao solo.

Analisar a correção do solo	Prescrição	Solicita ao usuário algumas informações e retorna a correção que deve ser feita no solo.
-----------------------------	------------	--

4.4.2 – Diagrama de Classes

O diagrama de classes foi dividido em cinco diagramas apenas por uma questão de facilitar a visualização, já que se trata de muitas classes e associações entre elas.

Os quatro primeiros diagramas mostram a hierarquia das classes, algumas dependências e associações. No quinto diagrama são mostradas as associações entre as classes, constituindo um grande emaranhado de relacionamentos necessários para a resolução do problema em questão.

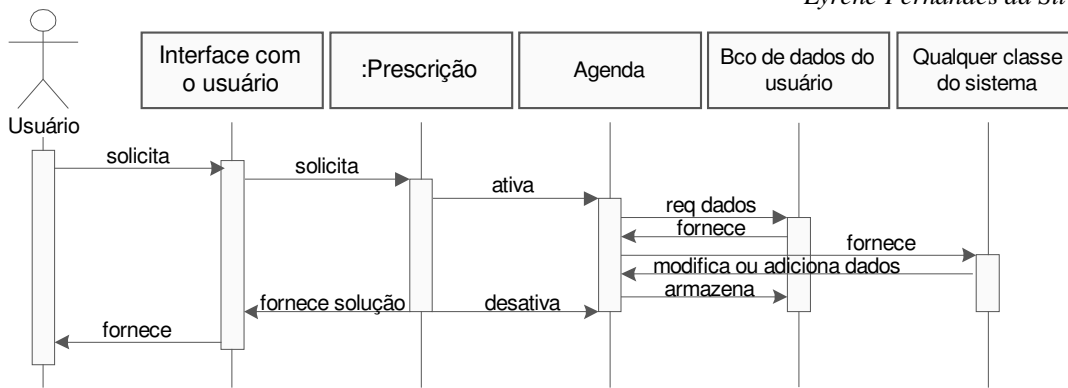


4.4.3 – Diagrama de Interação

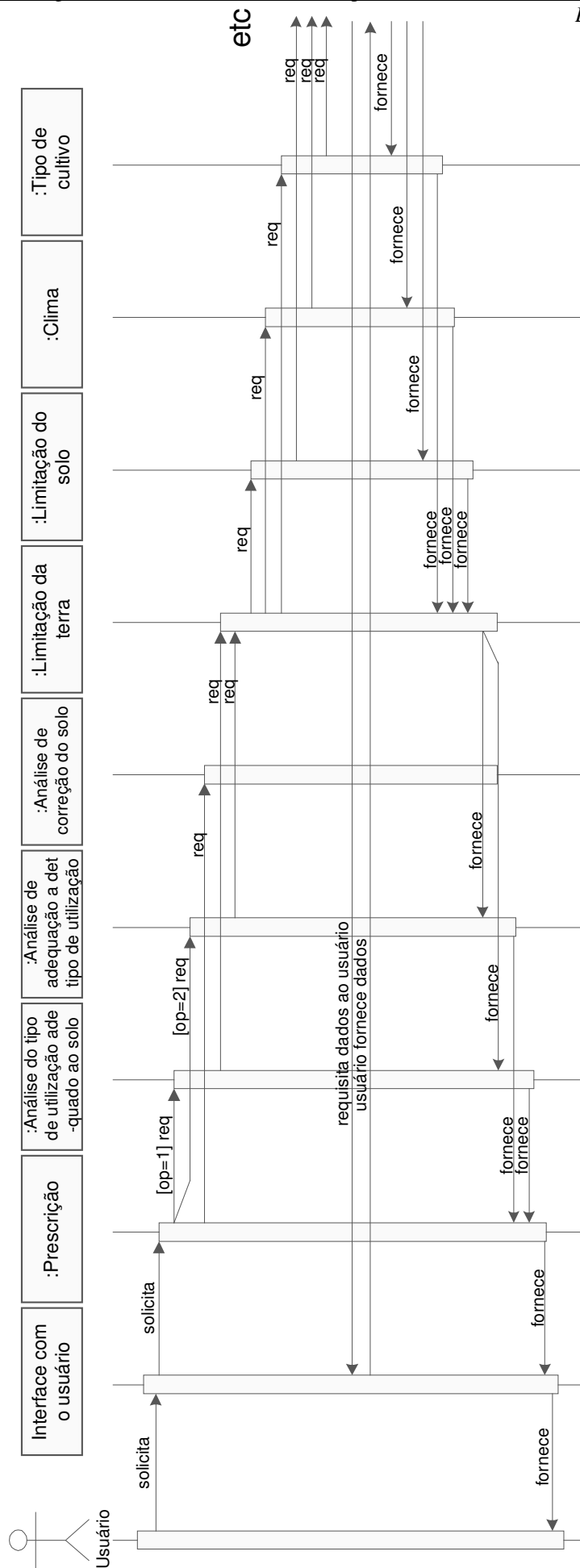
Os diagramas de interação foram construídos para o caso de uso “Requisitar a Prescrição”. Este caso de uso utiliza os outros casos de uso, a interação entre “Requisitar a Prescrição” e os outros é mostrada nestes diagramas. Tanto o diagrama de seqüência quanto o diagrama de colaboração não foram desenvolvidos totalmente pois, há uma grande quantidade de interações entre as classes, o que torna o diagrama incompreensível e muito extenso.

4.4.3.1 – Diagrama de Seqüência

Neste primeiro diagrama de seqüência é mostrado a interação entre a classe Prescrição e o banco de dados. Indica que ao ser solicitada uma prescrição, o sistema deverá ativar sua agenda e obter do banco de dados as informações armazenadas anteriormente em uma outra consulta. Isto significa que no decorrer da prescrição as classes só irão requisitar informações a outras classes se estas informações não estiverem na agenda.



No segundo diagrama de seqüência é mostrada a interação entre as classes do sistema a partir da prescrição e a linha de vida de cada uma delas. Há um grande encadeamento entre as classes. Elas requisitam informações às outras classes e aguardam resposta, para também fornecer informações às classes que as inicializaram. Embora o diagrama não esteja completo, ele dá uma visão de como normalmente as interações ocorrerão.



4.4.3.2 – Diagrama de Colaboração

4.4.4 – Diagrama de Estados

Foi construído um diagrama de estado para cada classe do sistema. Cada diagrama de estado representa os possíveis estados da classe e os eventos que devem ocorrer para que a classe mude de estado. Neste sistema os estados das classes são idênticos. Trata-se sempre de estar Requisitando alguma coisa, Recebendo algum dado, Inferindo algum resultado, e Fornecendo o resultado para quem inicializou a classe.

Sendo assim serão apresentados aqui apenas alguns diagramas de estados. O restante dos diagramas poderão ser consultados no Apêndice C.

Diagrama de estados para classe Análise de adequação a determinado tipo de utilização.

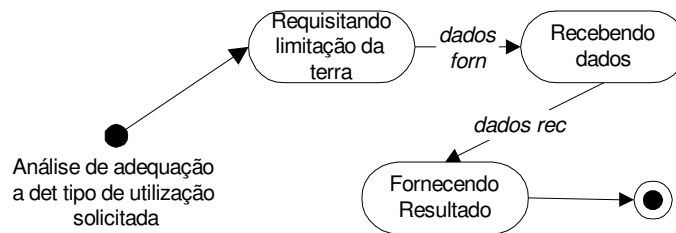


Diagrama de estados para classe Análise de correção do solo

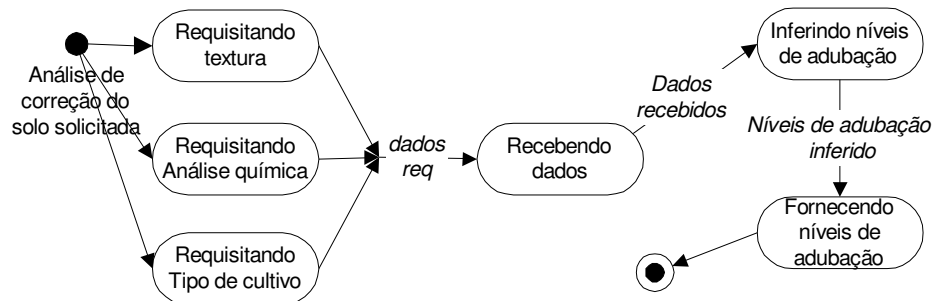


Diagrama de estados para classe Limitação da terra

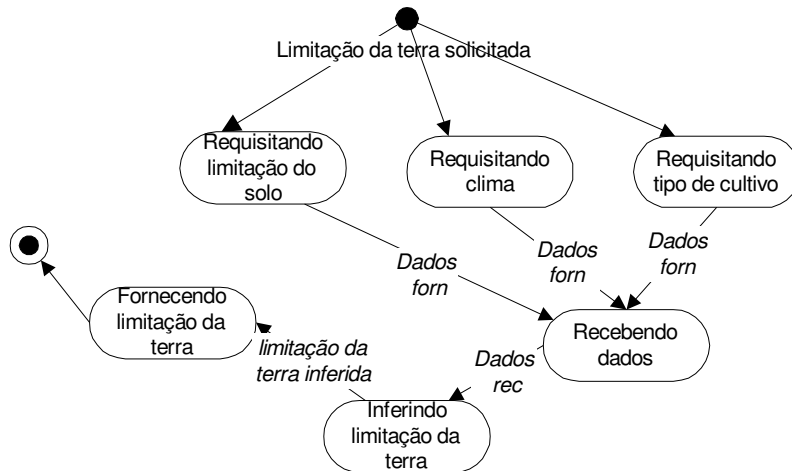


Diagrama de estados para classe Limitação do solo

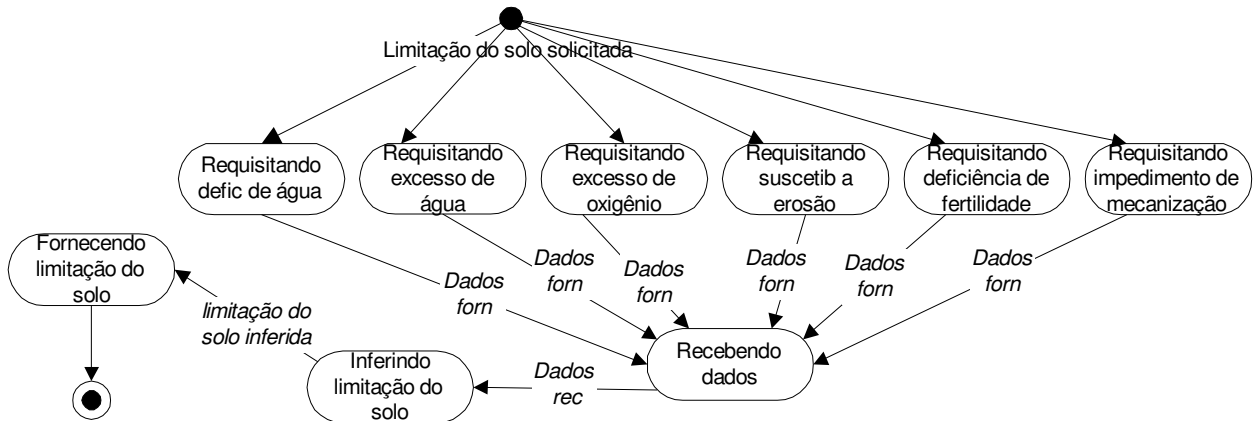
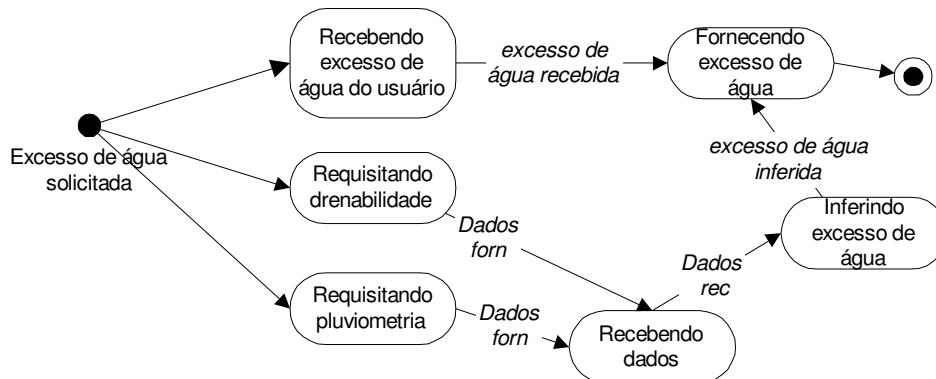


Diagrama de estados para classe Excesso de água solicitada



4.4.5 – Diagrama de Atividades

Foi desenhado um diagrama de atividades para cada classe. Estes diagramas identificam a seqüência de atividades realizadas no interior de cada classe quando são inicializadas. O diagrama de atividades captura bem este aspecto do sistema.

As atividades realizadas por cada classe são idênticas, se trata basicamente em Requisitar alguma informação a outra classe, Receber esta informação, Inferir algum resultado, e Fornecer o resultado obtido à classe que a inicializou. Alguns dos diagramas de atividades se encontram posteriormente, estando o restante no Apêndice C.

A atividade Agendar significa que uma vez calculado um certo resultado, este será inserido na agenda para utilização das outras classes, não sendo mais necessário calculá-lo novamente. Após toda a inferência realizada, os dados que estiverem na agenda são armazenados no Banco de dados para uma posterior consulta.

Diagrama de atividades para classe Prescrição

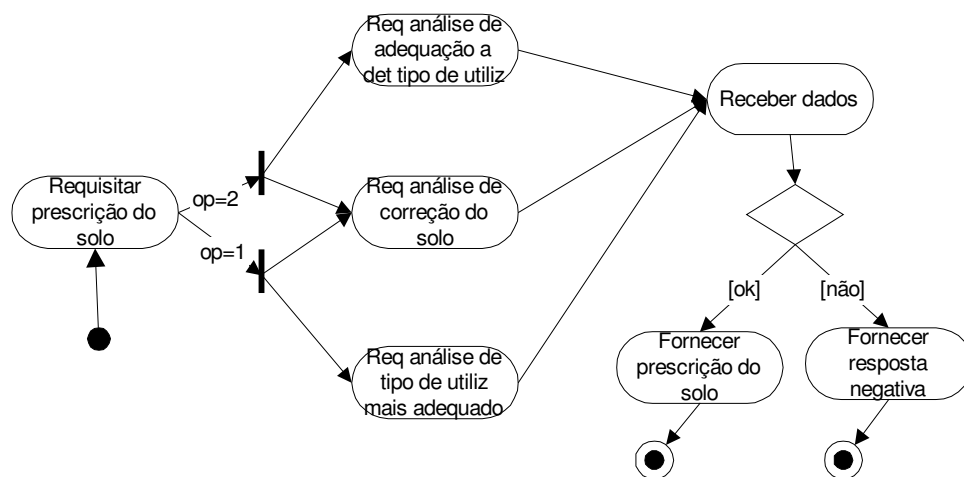


Diagrama de atividades para classe Análise de adequação a determinado tipo de utilização.

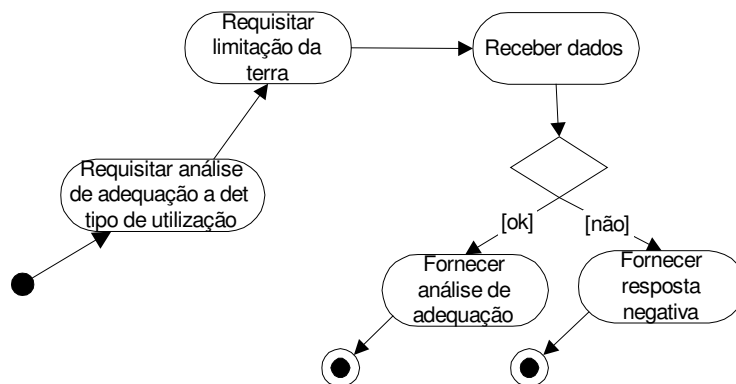


Diagrama de atividades para classe Análise de correção do solo

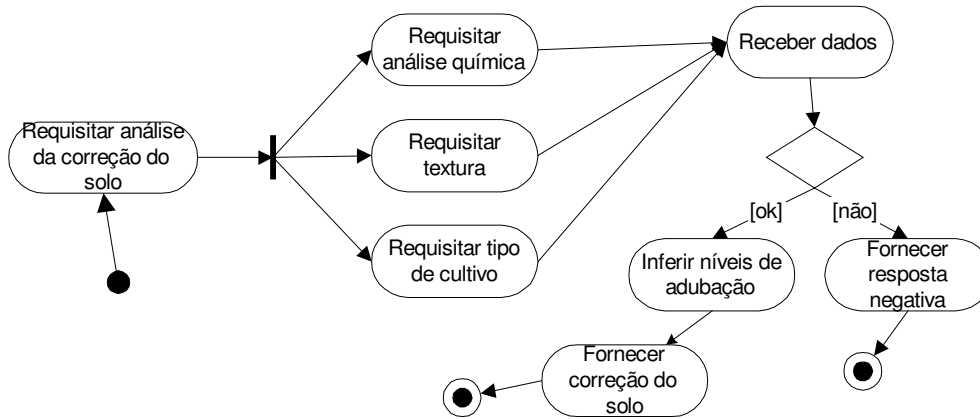


Diagrama de atividades para classe Limitação da terra

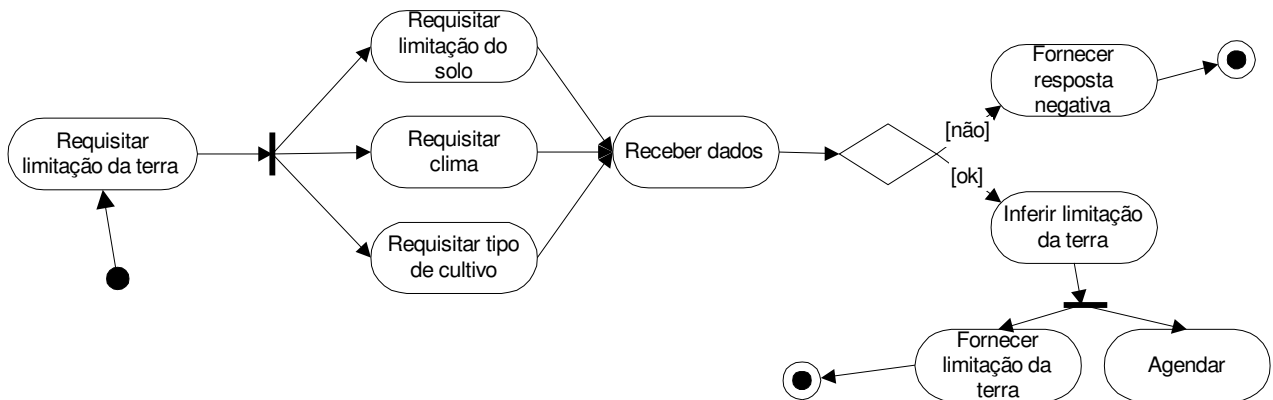


Diagrama de atividades para classe Limitação do solo

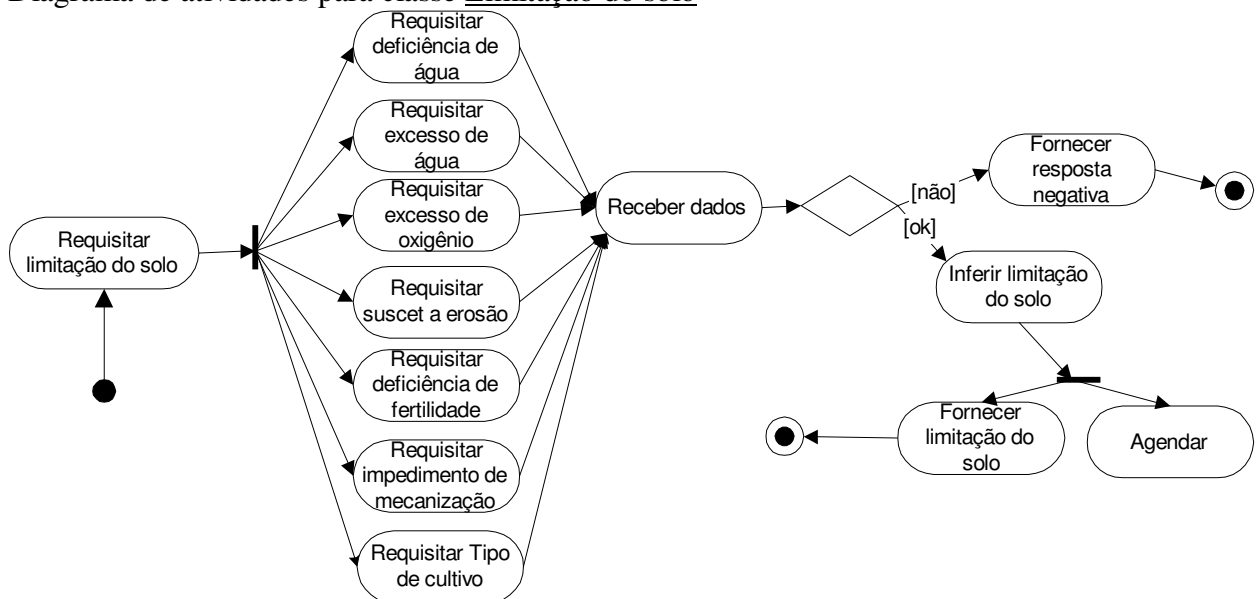
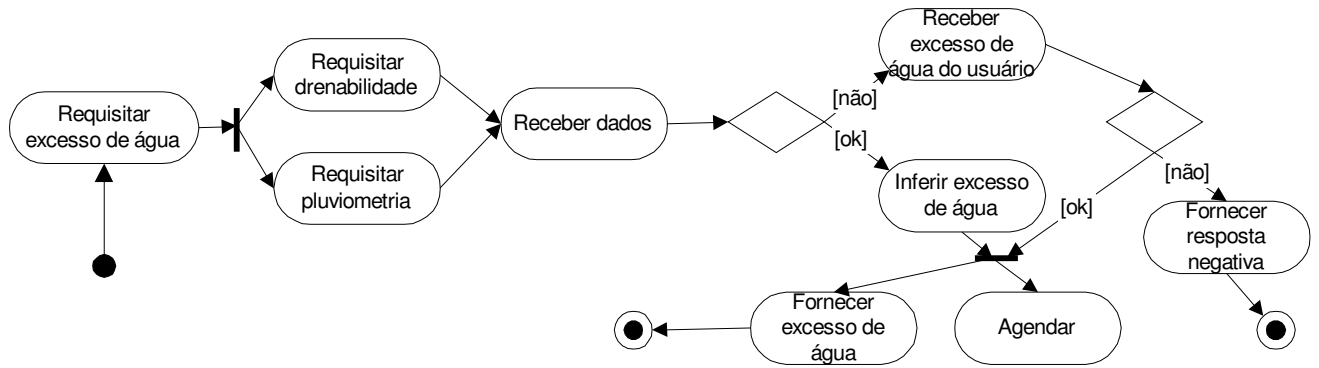


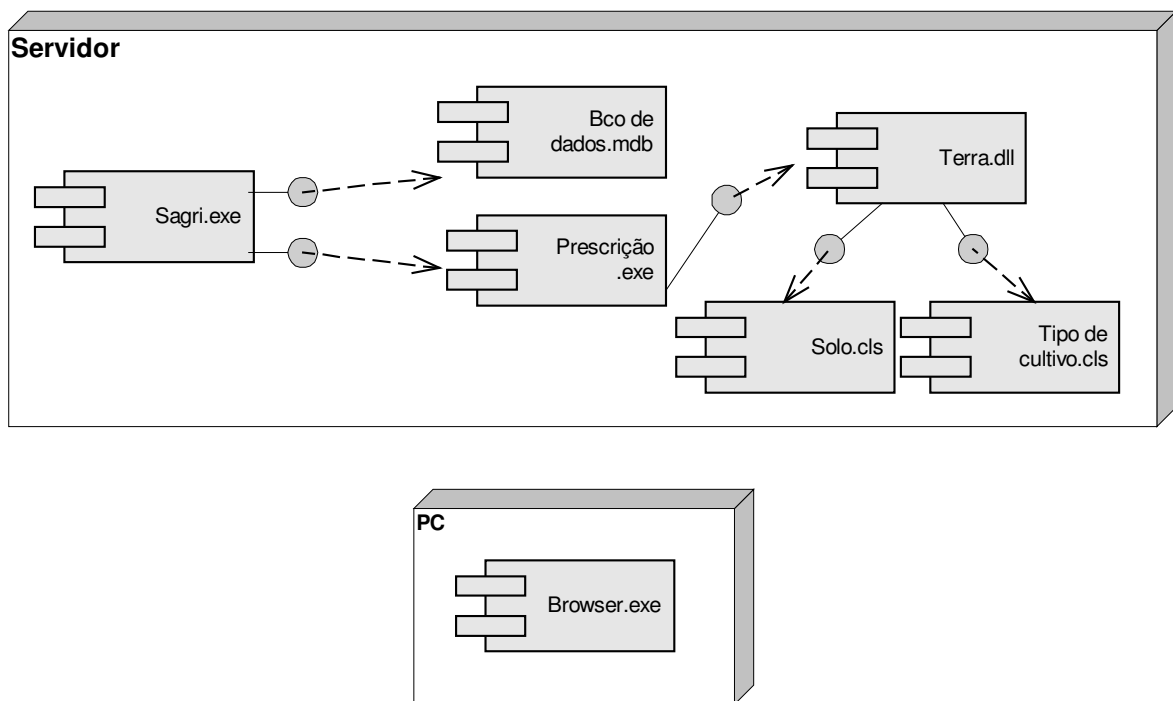
Diagrama de atividades para classe Excesso de água solicitada



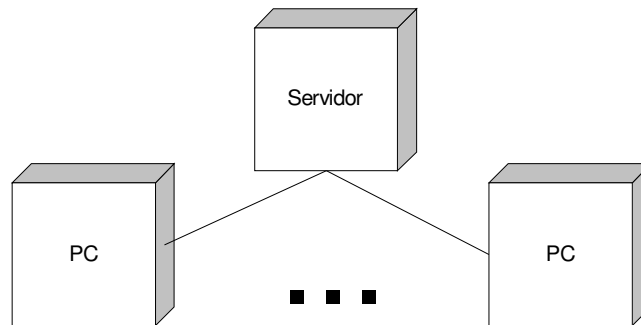
4.4.6 – Diagrama de Implementação

Os diagramas de implementação (componentes e execução) foram desenvolvidos de uma maneira simplificada, visto que o contexto a que se detém este estudo de caso é a análise e o projeto de software.

4.4.6.1 – Diagrama de Componentes



4.4.6.2 – Diagrama de Execução



É importante deixar claro que, para o desenvolvimento de um sistema utilizando a UML não é necessário que sejam desenhados todos os tipos de diagramas, o desenvolvedor deve se dedicar àqueles que se ajustam aos objetivos do sistema. Neste caso de estudo, foram desenvolvidos todos os tipos de diagrama por que o objetivo era avaliar a abrangência da UML em retratar os aspectos do sistema SAGRI.

5 – Conclusões Obtidas com as Modelagens

Como foi visto, a UML e a modelagem conceitual são métodos para modelar que se propõem a retratar o mundo real de forma completa, expressiva e correta, objetivando a implementação de sistemas de melhor qualidade. No entanto, o estudo teórico e o estudo de caso realizados, permitiram identificar vários fatores que diferenciam os dois métodos. São eles:

- ✓ Ambos os métodos baseiam-se no paradigma conceitual, onde o elemento principal é o conceito. Entretanto, a UML trata os conceitos como objetos, ou seja, estruturas constituídas de dados (atributos) e métodos (operações) que os manipulam. Enquanto que a modelagem conceitual trata os conceitos como conceitos, ou seja, ela é verdadeiramente conceitual, conceitual em sua essência. Os conceitos descritos no modelo conceitual também exercem ações, porém estas ações não constituem o corpo do conceito, elas se encontram distribuídas nas relações existentes entre os conceitos.
- ✓ Ambos utilizam elementos gráficos para representação do mundo, o que proporciona uma visualização menos complexa dos sistemas e conseqüentemente um ótimo meio para comunicação entre projetistas e entre estes e os usuários. No caso da UML, a utilização de objetos tornou a comunicação entre projetistas e pessoas de negócio mais fácil já que estes pensam em termos de eventos, objetos e políticas de negócio que descrevem o comportamento dos objetos, enquanto que, o modelo conceitual, por ser uma representação mais semelhante aos modelos mentais dos indivíduos, é mais natural ao ser humano, sendo desta forma mais facilmente compreensível por qualquer pessoa, expressando o conhecimento sem se preocupar com formalismos e notações.
- ✓ Nenhuma das duas modelagens possui uma metodologia explícita e bem definida. Porém necessitam adota-la para que se possam desenvolver modelos efetivos em satisfazer o objetivo da modelagem. Em analogia à maneira das pessoas desenvolverem seus raciocínios, pode-se optar por uma “metodologia” com princípios evolucionários onde a idéia principal é iniciar os modelos de maneira simples, tornando-os mais complexos em função do conhecimento de posse e da utilização. Seguindo este raciocínio, a UML induz o projetista a desenhar primeiramente o diagrama de casos de uso e o de classes simplificado (sem as operações) e depois refiná-los sucessivamente utilizando os diagramas de interação, de estados e de atividades.
- ✓ A modelagem conceitual pode ser conduzida iniciando-se do modelo dinâmico ou do modelo estático, dependendo do conhecimento e do tempo disponíveis para desenvolvê-lo e do modelo mental dos desenvolvedores do projeto (suas preferências por uma ou outra abordagem). Enquanto que a UML conduz o processo de desenvolvimento de uma maneira orientada a funções, ou seja, em analogia ao modelo conceitual, o modelo Orientado a Objetos seria uma espécie de modelo dinâmico, onde os diagramas vão sendo desenhados de forma a representar como e quando as operações serão realizadas para atingir o objetivo do sistema. A UML também possui modelos para representação do aspecto estático do sistema (diagrama de casos de uso e diagrama de classes) porém, estes modelos representam, da maneira mais compacta possível, as classes que serão tratadas no sistema. Enfim, enquanto na modelagem conceitual o projetista define o objetivo do

sistema e vai elaborando um estudo sobre os aspectos, os conceitos e os relacionamentos envolvidos, de forma que o modelo conceitual evolui paralelamente à estrutura cognitiva (referente ao sistema em questão) do projetista; para se construir um modelo em UML é necessário que o projetista já tenha uma estrutura cognitiva bem elaborada sobre o sistema a ser desenvolvido.

- ✓ O enfoque destes métodos é distinto. Enquanto a Modelagem Conceitual preocupa-se em modelar o mundo real para realizar estudos e análises de um mundo delimitado, a preocupação da UML é completamente implementacional. Esta característica coloca a UML em um nível diferente da Modelagem Conceitual. Uma ocupa espaço nos modelos lógicos, a outra nos modelos conceituais. De acordo com o ciclo de vida sugerido pela KADS, os Modelos Conceituais auxiliam os Modelos Lógicos no processo de desenvolvimento de sistemas.
- ✓ Ao utilizar a UML tem-se como pré-requisito o desejo de implementar o sistema usando a orientação a objetos. Para isto é necessário que se tenha realizado preliminarmente uma análise para avaliar se esse é, realmente, o formalismo de representação mais adequado ao sistema que se deseja implementar. Por outro lado, o Modelo Conceitual é completamente dissociado a qualquer mecanismo lógico de implementação. Ele oferece subsídio para que o desenvolvedor se familiarize com os conceitos que irá manipular, além de conhecimento sobre o contexto e análise do processo de solução mais adequado. Desta forma, pode-se utilizar o modelo conceitual como ferramenta para o desenvolvimento de sistemas computacionais ou não- computacionais.
- ✓ É possível conduzir a Modelagem Conceitual à níveis de detalhamento que se permita implementar o modelo diretamente pois, o Modelo Conceitual é construído de forma a abstrair e detalhar os conceitos e relacionamentos tanto quanto se deseja, obtendo meta conhecimento e instâncias, respectivamente. Tal condução dependerá apenas de dois fatores: da estrutura cognitiva do modelador, que tende a “crescer” (no sentido de se tornar cada vez mais bem elaborada e ser capaz de inferir mais conhecimento) a medida que se modela; e dos objetivos que se quer atingir: se é desejável detalhar o máximo possível ou se o desejável é chegar a um nível intermediário capaz de dar suporte a construção de modelos lógicos bem expressivos, como por exemplo, modelos orientados a objetos.

5.1 - Modelagem em UML a partir do mundo real

A modelagem em UML a partir do mundo real é, provavelmente, mais difícil pois precisa de um maior esforço mental por parte dos projetistas, a fim de mapear e simular na mente o sistema a ser desenvolvido. Para um sistema relativamente complexo esta hipótese é inviável já que a grande quantidade de conceitos e associações entre conceitos fazem com que se perca a visão do sistema como um todo ou mesmo de parte do sistema.

5.2 - Modelagem em UML a partir do modelo conceitual

A modelagem em UML a partir do Modelo Conceitual é extremamente facilitada, visto que, todo o ambiente do sistema já foi mapeado e analisado, as inconsistências, incertezas e erros do conhecimento adquirido identificados e “tratados” resultando num modelo de

conceitos e num modelo de solução bem planejados. Estes modelos são transformados em modelos O.O. sem que seja necessário grande esforço dos projetistas. O diagrama de classes é facilmente construído a partir do modelo estático e os diagramas de casos de uso, interação, estados e atividades desenvolvidos tendo como base o modelo dinâmico, apoiado no modelo estático e, é claro, auxiliado enormemente pela evolução da estrutura cognitiva do modelador.

6 – Conclusões e Projetos Futuros

A Orientação à objetos é de fato a tecnologia em ascensão no momento. Ela oferece suporte desde à fase de análise e projeto, no desenvolvimento de sistemas computacionais, com os métodos de Booch, OMT e OOSE entre outros, até a fase implementacional com as linguagens orientadas a objetos, visuais ou não.

A utilização da UML firma, ainda mais, a Orientação a Objetos na fase de análise e projeto, fornecendo recursos padronizados (diagramas) para modelagem de sistemas computacionais. Entretanto, ainda é necessário recursos que permitam expandir o conhecimento do projetista, referente ao sistema a ser desenvolvido.

Verificou-se com este trabalho, que um Modelo Conceitual construído antes do modelo lógico orientado a objetos proporciona ganhos bastante significativos, como por exemplo, sistemas mais reais e soluções mais eficientes aos problemas, de forma a elevar a qualidade dos sistemas assim desenvolvidos. Além de oferecer subsídio ao projetista, para expandir e/ou modificar o sistema construído, tornando-o mais flexível e de fácil manutenção, entre outros.

A transformação do modelo conceitual nos diagramas da UML é bem gradativa, não é necessário um grande esforço por parte dos projetistas. O que leva a crer que o tempo supostamente perdido é recompensado pelo pequeno intervalo de tempo que se despenderá na modelagem em UML. Com a vantagem de que este modelo orientado a objetos será bem mais robusto, correto e eficiente.

O estudo realizado fornece um bom embasamento teórico sobre modelos conceituais, abordados como ferramenta a auxiliar o processo de desenvolvimento de sistemas de maneira efetiva.

Projetos Futuros

A Modelagem Conceitual (como foi abordada) é uma tendência tecnológica, porém, consiste em uma questão em aberto. Muito ainda se deve pesquisar sobre a melhor forma de construí-lo, a adoção de uma metodologia é extremamente necessária. Pesquisas em áreas correlatas com a cognição podem alavancar as pesquisas em informática pois, conhecendo-se melhor o processo de cognição humano, pode-se criar modelos artificiais que dêem suporte ao desenvolvimento de sistemas, como é o caso dos modelos mentais e dos mapas conceituais.

Projetar ferramentas que capacitem os analistas a editar (desenhar) o modelo e, talvez, até executá-lo, simulando o sistema a que se almeja construir, ou mesmo, sendo o próprio sistema. É um trabalho ainda visto como audacioso, porém a tecnologia avança cada vez mais rápido, e não mais que há trinta anos atrás a O.O. também era considerada como projeto “ambicioso”. Tais projetos logo estarão em desenvolvimento sob a orientação da professora Dr^a Márcia de P. B. Gottgroy.

7 – Bibliografia

- [1] GOTTGROY, Márcia P. B.; Câmara, G. et all (1995) SAGRI: Sistema Inteligente de Apoio à Produção Agrícola. Agrosoft'95, Juiz de Fora - MG, 1995.
- [2] GOMES, Magnus K. N. F.; Projeto de Interface do Sistema SAGRI Utilizando Sistemas de Informações Geográficas. Agrosoft'97.
- [3] OLIVEIRA, Marcos Antonio; Implementação do Raciocínio Aproximado no Sistema Sagri (Sistema Inteligente de Apoio a Produção Agrícola), Relatório de Graduação, UFRN, Natal-RN, 1997.
- [4] OLIVEIRA, Marcos A.; A Importância do Tratamento do Conhecimento Impreciso no Desenvolvimento de Sistemas para a Agricultura. Agrosoft'97.
- [5] LOPES, Maximiliano Araújo da Silva; Uma Abordagem Preliminar sobre Modelagem no Desenvolvimento de Sistemas Computacionais, Relatório de Graduação, UFRN, Natal-RN, 1998.
- [6] GOTTGROY, Márcia P. B.; Relatório Técnico do Projeto Coopera – Ambiente de Aprendizado Cooperativo. UFRN, Natal-RN.
- [7] PRESSMAN, Roger S.; Engenharia de Software, Makron Books.
- [8] GOTTGROY, Márcia de P. B.; O Processo de Aquisição do Conhecimento na Construção de Sistemas Especialistas, *em Engenharia de Sistemas e Computação*, pela COPPE/UFRJ, concluído em 16.03.1990.
- [9] GOTTGROY, Márcia de P. B.; Sistemas Baseados em Agentes- Capítulo do Livro da I Escola Regional de Informática RJ/ES – ERI/SBC – de 23 a 27/03/1998 – Rio de Janeiro e Espírito Santo.
- [10] GOMES, Apuena V.; A Aplicação da Metodologia KADS na Construção de Assistentes Inteligentes. UFRN, Relatório de Graduação 1997.
- [11] CUNHA, Fernanda dos Santos; Aquisição de Conhecimento.
- [12] MOREIRA, Marco Antonio; Modelos Mentais, trabalho apresentado no Encontro sobre Teoria e Pesquisa em Ensino de Ciência –Linguagem, Cultura e Cognição, Faculdade de Educação da UFMG, Belo Horizonte, 5 a 7 de março de 1997.
- [13] BORGES, T; Modelos Mentais. XII SNEF - Belo Horizonte, 1997.
- [14] Representation de la connaissance.
<http://tecfa.unige.ch/staf/staf9597/beltrame/STAF11/concepts.html>

- [15] CARDOSO, Carlos Adriano; *Cognição, Modelos Mentais e Teoria do Conhecimento: Notas para um Estudo das Representações Mentais dos Usuários de Computadores.*
- [16] MOREIRA, Marcos A. & MASINI, Elsie F. Salzano; *Aprendizagem Significativa – A Teoria de David Ausubel*, ed. Moraes, São Paulo, 1982.
- [17] AZEVEDO, Douglas José P.; *Modelo Conceitual do Sistema de Matéria para Publicações*, Bate Byte.
- [18] DILLON, T. S. & TAN, P. L.; *Pretice Hall, Object Oriented Conceptual Modelling*, 1993.
- [19] OLIVEIRA, José Palazzo M.; *Modelagem e Projeto de Sistemas de Informação: Organizações Virtuais.* <http://www.inf.ufrgs.br/~palazzo/em/>
- [20] MARTIN, James & ODELL, James J.; *Análise e Projeto Orientados a Objetos*, ed Makron Books, São Paulo, 1995.
- [21] FURLAN, José Davi; *Modelagem de Objetos através da UML – The Unified Modeling Language*, São Paulo, Makron Books, 1998.
- [22] PALLAZO, L. A. M. ; *Aspectos da Modelagem de Sistemas de Informações Inteligentes.* UFRS, Tese de Doutorado, 1996.
- [23] BARROS, Pablo Fernando do Rêgo; *UML - Linguagem de Modelagem Unificada - Aracaju-SE: Universidade de Tiradentes, Monografia.*
- [24] RUMBAUGH, James, BLAHA, Michael, PREMERLANI, William, EDDY, Frederick & LORENSE, William; *Modelagem e Projetos Baseados em Objetos - ed. Campus*, 1994.
- [25] RATIONAL; *UML Sumary, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [26] RATIONAL; *UML Semantics, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [27] RATIOANL; *UML Notation Guide, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [28] RATIONAL; *UML Extension for Objectory Process for SoftWare Engineering, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [29] RATIONAL; *UML Extension for Business Modeling, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [30] RATIONAL; *Object Constrain Language Specification, versão 1.1, 09/1997.* <http://www.rational.com/uml>
- [31] DANTAS, José Araújo; *Caracterização Edafo-Climática e Classes de Aptidão Agrícola da Fazenda Lagoa Nova/Furnas, Município de Riachuelo –RN*, 1997.
- [32] PIMENTEL, Mariano Gomes; *Modelo Orientado a Conceitos (MOC)*, <http://www.lia.ufc.br/sbie98/anais/artigos/art23.html>

- [33] SILVA, Lyrene Fernandes; Utilização da Modelagem Conceitual no Desenvolvimento do Sistema Sagri, apresentado e publicado no I Workshop em Lógica, Banco de Dados e Inteligência Computacional, realizado nos dias 09 e 10 de novembro de 1998, na UFRN.

- [34] STRACK, Jair; GPSS: Modelagem e Simulação de Sistemas, Rio de Janeiro; LTC Ed S.A., 1984.

- [35] MORTIMER, Eduardo Fleury; Construtivismo, Mudança Conceitual e Ensino de Ciências: Para onde vamos?, apresentado no III Escola de Verão de Prática de Ensino de Física, Química e Biologia, realizada de 10 a 15 de outubro de 1994, em Serra Negra – SP.

- [36] SILVA, Lyrene Fernandes; Modelagem Conceitual como Ferramenta para o Desenvolvimento de Sistemas Computacionais, Monografia, UFRN – Natal – RN, 1999.